SCIENTIFIC PROGRAMME

INVITED SPEECH

LOOKING FOR INTELLIGENT RECONFIGURABLE SIMULATION

Tudor Niculiu[B], Anton Manolescu[B] and Sorin Cotofana[D]
Universitatea POLITEHNICA din Bucuresti, ETTI Dept.
Splaiul Independentei 313, 060042 Bucuresti, România
tudor-razvan@ieee.org
Delft University of Technology, EE Dept.
Mekelweg 42600, GA Delft, The Netherlands
S.D.Cotofana@ewi.tudelft.nl

KEYWORDS
Conscience, Hierarchy Types, Simulated Intelligence.

ABSTRACT

Simulation relates function to structure. Researching intelligence by simulating it to get to intelligent simulation demands the study of essential abstract structures: the human mind, the different types of hierarchies, and the simulation as relation between static and dynamic structures. Intelligence and Faith, as any other dichotomies, can converge together to integration, or can destroy each other if they are not associated. Conscience is the link. Both intelligent simulation and the simulation of intelligence demand transcending the present limits of computability toward simulability by an intensive effort on extensive research to integrate essential mathematical and physical knowledge guided by philosophical goals. We extend the reconfigurability to the simulation itself. First, by a self-aware simulation, for that we build a knowledge hierarchy corresponding to the simulation hierarchy, we get a self-control of the simulation process. Then, by expressing both simulation and knowledge hierarchies in the reference system of the basic hierarchy types (classes, symbols, modules), which is derived from the main partition of our Real Life (Beauty, Truth, Good), we aim to create the context for a self-organization of the simulation. So we try to model the conscience for simulating the intelligence, and then to reach for intelligent simulation.

ARGUMENT

Faith-Intelligence are ☯ in our life (Way, Truth, Life).

Intelligence = (consciousness, adaptability, intention) and *Faith* = (inspiration, intuition, imagination) are complementary parts of the human mind, separated by *Conscience* = (consciousness, inspiration).

Reality is not confined to *Nature*, as the cardinal of the discrete IN is less than the one of the continuous IR, but *Reason*, which is natural, as |IN| = |Q|, is dense in the Reality, as IR is the analytical closure of the (discrete) rationales. This shows that neither pure reason-based adaptability nor pure intuition can approach Reality without being integrated by conscience and communicating by intention and imagination.

The natural limit of complexity is caused by the essentially sequential approach, whereby the real limit of computability results from the discreteness of our reason, when considered context-free in our mind. The first idea is to consider/ remember that reality is more than nature, as the continuum of IR is more powerful than the discrete universe of IN. The second analogy is that integer beauty is not enough to comprehend the Reality. The third argument is that reason is less than our real thoughts, as the cardinal of |Q is $\aleph_0$.

Although |Q is dense in |R, so pure reason could converge to reality, the complexity problem limits the computability. The essential limit of the discrete computability, as of the computable intelligence, results from the self-reference, demanded by the integration of level and metalevel needed for consciousness. A hierarchical type is necessary to represent conscious knowledge.

Even if for the moment other essential aspects of the faithful intelligence can neither be constructive or intuitive, they should not be neglected. For example, there are much more real things than those reasonably imagined, although between any two real numbers there is a rational one (not intuitive). And we know that if there is no cardinal between that of the countable sets and that of the continuous ones, then there exists no other logical value than true and false, what simply hurts the human in his love for nuances. This can be avoided only if we believe (not constructive) that an intermediary level between Natural Reason and Reality exists, as the wises think (Plesu 2003):

There are angels between humans and God.

The historical experiment of the pure reason had sense in our evolution, but should have ended two centuries ago, when geniuses, as Beethoven, Byron, Hegel, Gauβ, Goethe, Kant, Mozart, Napoleon, Schiller, more synchronized than ever, proved the power of integrating

- faith with intelligence by conscience,
- art, science and construction by metaphysics,
- human and humanity, by their individual work that no one thinks to surpass alone.

Human thoughts can not be explained or handled by adaptability-based reason, even if this masters non-determinism or parallelism.

Reason has to extend to intelligence in the context of faith. An intuitive way is to integrate consciousness, then intention and imagination to intelligence, and then to extend the research towards inspiration and intuition. The power of abstraction is the real measure for the human mind. Turning abstraction into comprehensive construction could be the aim of humanity, the unique *God* for different cultures of free humans.

Freedom is understood necessity

Georg Friedrich Wilhelm Hegel reminded us of the real sense of freedom, so we have to recall our conscience to reintegrate our mind and to remember that society has to assist humans to live among humans, not to consider that humans just have to work for the society. An operating system serves the autonomous programs, both for the function of the hardware and for development of the software. The society has to assure health and education for everyone, and encourage search and research for any conscious human. Society is only the memory of the past, and the manager of the present problems to live together in respect of the others on the way to understand each other, building us together to more essential beings for an integrated existence.

RECONFIGURATION

Applying *Divide et Impera et Intellige* to hierarchy types reveals their comprehensive constructive importance based on structural approach, symbolic meaning, object-oriented representation. Formal hierarchical descriptions contribute to a theoretical kernel for self-organizing systems. A way to begin is hierarchical simulation. A way to confirm is the object-oriented reconfigurable simulation.

Reconfigurable computing architectures complement existing alternatives of spatial custom hardware and temporal processors, combining increased performance and density over processors, with flexibility in application. The experimented ways to reconfigurable design are Field-Programmable Gate Arrays for circuits (Rabaey 1997) and reconfigurable networks for systems (Miller et al. 1993).

Our project (Niculiu and Cotofana 2003) extends the reconfigurability to the simulation itself. Intelligent self-organization needs consciousness to control adaptability for reconfiguration. Towards a self-aware simulation to control the simulation process we build a knowledge hierarchy corresponding to the simulation hierarchy, and by expressing both simulation and knowledge hierarchies in the reference system of the basic hierarchy types we create the context for a self-organization of the simulation. We try to reach this goal integrating hierarchical intelligent simulation into nanotechnological realization (Goldstein and Budiu 2001)

Recursive reconfiguration of the simulation process, at any hierarchy level, is allowed by different strategies that alter one of the technique/ model/ method if one of the imposed properties is not fulfilled after applying a technique, using a model and suitable methods for evaluation and reconfiguration. The process repeats for the initial description or the one resulted from prior (insufficient) improvement. This calls for an intelligent control system that assists/ automates the reconfiguration. The techniques use hardware-software model templates, whose methods are recursively handling the different components in the system's description. Measurement functions control the continuation process of the reconfiguration, what suggested bringing reconfiguration in the context of software and hardware, as the strategies can be expressed object-oriented/ categorical and developed/ understood mathematically.

HIERARCHICAL SIMULATION

The research on cosimulation inspires the study of essential abstract structures: human mind, different hierarchy types, and simulation - as relation between static and dynamic structures, or even, at a higher abstraction level, between structure and function (Niculiu and Cotofana 2004). Towards this goal we put in correspondence three triplets of concepts of different collaborating domains: *hierarchy types* (class, symbol, structure), simulation abstractions (syntax, semantics, pragmatics), basic philosophy: (Beauty, Truth, Good). More points of view can confirm a selection of the essential things to begin marching on the way. Further necessary correspondences to approach and goal partitions of the human evolution are explained.

Simulation = (representation, goal).

Representation is a 1-to-1 mapping from the universe of systems (objects of simulation) to a hierarchical universe of models (a representation can be inverted). A model must permit knowledge and manipulation, so it has two complementary parts/ views: description and operation. In a formal approach models correspond to classes and specifications to instances.

Understanding and construction should use correspondent hierarchy types, i.e., a reflexive kind of abstraction has to be expressed by the *knowledge hierarchy type*. Knowledge and construction hierarchies cooperate to integrate design and verification into simulation; object-oriented concepts are symbolized to handle data and operations formally; structural representation of behavior manages its realization. A hierarchical approach is needed to handle both knowledge and metaknowledge. Hierarchy types open the way to simulate intelligence as adaptable consciousness by integrating the system and the metasystem. Hierarchy is the syntax of abstraction.

Hierarchy is a network that can represent any mathematical structure type (algebraic, topological, order). Hierarchies are leveled structures, which represent different domains. A level is an autonomous mathematical structure, containing abstract/ concrete entities, linked by level scoped relations.

Abstraction relates the levels: this induces an order relation between levels, partial, concerning entities, and total, regarding the levels. Beyond the hierarchical point of view, the system can be formalized as an autonomous domain, structured by metahierarchical relations, to build a level in a higher order hierarchical system. Hierarchical structures exhibit two complementary processing strategies: top-down and bottom-up. Coexistent interdependent hierarchies structure the universe of models for complex systems, e.g., hardware/ software ones. They belong to different hierarchy types, defined by simulation abstraction levels, autonomous modules, classification, symbolization and different grades of knowledge abstraction.

Abstraction and hierarchy are semantic and syntactical aspects of a unique fundamental concept, the most powerful tool in systematic knowledge. The (hierarchy, abstraction) concept is a particular form of Divide et Impera et Intellige: hierarchy results of formalizing abstraction. There are different kinds of abstraction that need different types of hierarchy. Most abstractions are simplifying, what is compulsory for complex object-systems. Correspondent to the construction/ simulation hierarchy type, a knowledge hierarchy type has to formalize reflexive abstraction in order to enable intelligent simulation.

Classes abstract the form, symbols the contents, and partitions simplify the approach. All these assist the simulation hierarchy to construct, verify, optimize and test, being managed completely by discrete formalisms/ simulations of the pure reason. The knowledge hierarchies promise to integrate discrete into continuous, on the way to extend pure reason to less constrained thinking. Hierarchies of different types correspond to the kind of abstraction they reflect (↑the abstraction goal):

- Class hierarchy (↑concepts) ↔ virtual framework to represent any kind of hierarchy, based on form-contents, modularity, inheritance, polymorphism.
- Symbol hierarchy (↑metaphors) ↔ stepwise formalism for all kind of (hierarchy) types.
- Structure hierarchy (↑strategies) ↔ stepwise managing of all (hierarchy) types on different levels by recursive autonomous block decomposition.
- Construction hierarchy (↑simulation) ↔ simulation (design/ verification/ optimization/ testing) framework of autonomous levels for different abstraction grades of description.
- Knowledge hierarchy (↑theories) ↔ reflexive abstraction, aiming that each level has knowledge of its inferior levels, including itself. This hierarchy type offers a way to model conscience.

Metaphor is a popular instance of abstraction. God is the absolute abstraction. And if we remember that liberty is understood necessity we can detail the metaphorical thesis:

God is the evolution goal of our faithful intelligence.

We can reduce abstraction to its simplifying types: classes, symbols, modules, and construction, hoping to get to the absolute liberty, i.e., considering God, the simplest item of the Reality, totally unconstrained. But we can simulate/ construct/ live/ work associating a knowledge hierarchy to everything we do, aiming to understand constructively the *most complex absolute necessity, defining God.* The power of abstraction is human's gift to surpass the natural limits, extending pure reason to real intelligence. As any other dichotomy pair, faith and intelligence can evolve convergent to integration, or can destroy one another if they are not linked together constructively.

Divide et Impera et Intellige has three parts as *alle guten Dinge sind drei.* Mathematics develops from three basic structure types, usually integrating them: algebra, order, and topology. We divided our existence in three collaborating parts: arts, sciences, and technology, correspondent to our world of beauty-loving ideas, our world of truth-searching efforts, and our (presently exaggerated) world of good-aiming constructions.

1. Mathematics - the most accessible art (Hofstadter 1979) - discovers and studies (Bourbaki 1966) types of structures: (algebra, topology, order), correspondent to (construction, orientation, understanding) as example of correct and complete integration, to be followed by science and technology. *Art is for art*, so it's defining itself, looking for the Beauty.
2. Physics - the paradigmatic science - should integrate its fundamental forces in a theory (Traub 1999), and all natural and social sciences - as chapters, leading them to really apply mathematics. Social sciences study a universe, as complex and non-deterministic as the natural one, so mathematics is at least as important to them as for natural sciences. An integrated science would also better inspire mathematics. Science raises the fear and the research inspired by it to more abstract domains, i.e., it is defined, as *Fear of God*, looking for the Truth.
3. Engineering has to be closely related to mathematical approach and integration of parts, not only to mathematical techniques, as to scientific courage and multiple views, not only to scientific results (Ciaffaglione and diGianantonio 2001). As reality contains the abstract ideas, even if physics could explain everything discretely, the power of continuum can not be forgotten; analog engineering should not be neglected in modeling and simulation. Paying attention only to the Good in our life, is most dangerous, as this part of the Reality, called *mental world* (Penrose 1989), is defined by its complement, so it is not better than this, if not closely constrained by Art and Science.

The classical activities in complex systems simulation that regard different levels of the construction or knowledge hierarchy, can be expressed symbolically, represented object-oriented and simulated structurally.

Constructive type theory permits formal specification and simulation, generating an object satisfying the specification. Complex simulation needs consistent combination of mathematical domains and an intelligent compromise between consistence and completeness. Intelligence simulation implies a hierarchical approach of different types. Any application of it can be imagined as an educational system to discover models for conscience and understanding. The formalism for hierarchy types is the category theory (Ageron 2001). Different domains permit a unified formalization in the theory of categories, and a unified representation using object-oriented templates. Simulation should remain correct, with extended requirements for the object-system, regarding complexity, optimization and (sequential/ parallel) competence for different domains. The hierarchical principle, applied to knowledge and simulation, (locally) bounds the complexity, by problem decomposition, and assures (almost) correct-by-construction design and efficient (design-adapted) verification.

HIERARCHICAL COSIMULATION

Cosimulation of coexistent domains is an important step for collaborative specialization, the next step to Intellige after Divide et Impera and an essential need before approaching conscience modeling. Testability is the technological correspondent of sincerity, which is essential for intelligence and communication.

Hardware-Software Cosimulation

The hardware-software cosimulation of complex systems is imposed by the lack of compatibility or optimality associated with the initial hardware/ software partition of a design, and by the inefficiency of the design-verification cycle in the context of a fixed partition (Niculiu et al. 2002). To unify simulation methodologies, we started from the results of different research directions: object-oriented hardware/ software description, formal verification of software/ hardware, automated synthesis of hardware systems. A unified representation for hardware and software allows techniques from one domain to be applied to the other domain. Therefore, a representation based on abstraction and object-orientation, used primarily for software, is employed for the hardware domain as well. Also, existing software techniques, such as those used for verification of abstract data type implementations, can be used for hardware.

Knowing the features (mandatory: abstraction, hierarchy, encapsulation, modularity, message passing + optionally: typing, concurrence, persistence) that characterize an object-oriented language, they also make sense from the perspective of hardware modeling and simulation. Object-oriented specification of models can be based on general systems theory, what makes this approach applicable in all domains. The designed framework permits self-organizing. It offers at any abstraction level of the simulation hierarchy: system description in a commonly used language extended for parallelism by synchronization items; automatic learning-based hardware/ software partition of the description; consistent communication between heterogeneous parts and with the exterior; simulation of the whole system during any design phase. Data abstraction can be used to represent hardware. A class corresponds to a set of elements with common static and dynamic characteristics. Thus, a hardware component can be treated as class containing state along with a collection of associated operations that can manipulate this state. For example, a register can be viewed as a class with the operations read and write. The contents of the register correspond to its state, which can be accessed and manipulated using the operations read and write, respectively.

Software engineering utilizes data decomposition to refine (derive implementations for) abstract data types. When modeled as data abstractions, hardware elements can also be refined using this decomposition technique. Generic types result from the ability to parameterize with types a software element, such as procedure or data type. This makes programs more general. The template concept, that realizes it in C++, can be applied to hardware components that act as containers, e.g., registers, register files.

Digital-Analog Cosimulation

The essential difference between analog and digital simulation paradigm is induced by that between the mathematical structures their models are based on algebraic for digital versus analytical for analog. In view of intelligent simulation the whole intelligence has to be simulated, i.e., conscience along with adaptability. The discrete parts of simulation, e.g., a sequence of decisions/ stimuli for simulation, do not easily match the continuity of the analog part.

Usually, the difficulty of analog simulation is avoided by defining an auxiliary representation domain, intermediate between the behavioral and the structural, where the problem is decomposed into topology selection and dimension computation. The first process is discrete and the second one is continuous over a restricted problem space. Object-oriented representation lends itself for this complementary form-content instance.

But, topology selection would be more systematic if continuous modifications of the form were possible, and dimension computing is more efficient if symbolic algebraic methods are used. We searched the compromise between simulation algebra and analog analysis in three directions: defining upper levels of abstraction for the algebraic laws governed analog, modeling analog simulation in algebraic-analytical structures (of functional analysis) or association of analytical syntax to the analog simulation process (Zhong and Weihrauch 2003). All these approaches are suited to object-oriented *Analog Hardware Description Language* (AHDL).

Thermal-Analog Simulation

The development of Computer Aided Design (CAD) procedures for microsystems imposes the simulation of thermal phenomena as secondary effects to the main, analog - electronic, mechanical, optical, or chemical - ones. As the microsystem components are modeled in an AHDL the models can be enhanced with temperature dependence and power generation estimation. Moreover models for environment and packaging conditions can be added as well. AHDL models permit direct simulation of the microscopic thermal transfer, and qualitative simulation-oriented representation of second order effects. Consequently, different physical domains, described by isomorphic analog laws, can be simulated in a unique representation (Zeigler et al. 2000).

Electro-, hydro-, thermodynamics, or circuit theory can be expressed with through-across concepts governed by dual topological laws for continuity and compatibility. AHDL enables a direct physical simulation of heat conduction, alternative to discrete heat equation: if only the first order relation representing Fourier's hypothesis is expressed in the model, the integration and discretization are realized by topological constraints that characterize AHDL structures. This suggests the idea:

Simulation is computer-oriented theory.

Behavioral Adaptable Design for Testability

Design-for-testability (DFT) must suit the behavioral specification of today's complex system design. Referring to high-level synthesis, DFT can operate before, while or after it. The first choice permits the intervention of an intelligent agent for adapting the DFT technique, model or method to the particular design. We call it behavioral adaptable design-for-testability: it improves the testability, measured with adequate methods, direct on the behavioral specification or aided by special representations, that have to permit returning to the behavioral description after improving the testability of the system to be designed.

The results are general enough to be valid for systems, either hardware, software or hard/ soft. The most used DFT techniques are Scanning, Built-In Self-Test and Test Point Insertion. They can be applied at the different levels of the design hierarchy (behavior, Register Transfer Level - RTL, logic) and can be combined.

We began with Partial Scan applied to the autonomous blocks of the behavioral HDL specification, but the other techniques can contribute to improve the testability of the behavioral specification or the way to this goal. The Partial Scan problem is the selection of the scan registers following a strategy to find an optimum testability - complexity compromise; this is better shaded by analog computing (Blum et al. 1998).

Memory elements - registers (arrayed flip-flops)/ flip-flops/ latches (non-clocked flip-flops) - are represented in behavioral hardware descriptions by variables or signals.

Variables are description objects local to processes/ subprograms, used to store intermediate values between sequential statements, characterized by free assignment.

Signals are permanent description objects to link concurrent elements: components/ processes/ concurrent assignments, demanding synchronized assignment, declared locally - within architecture, block or other declarative region, or globally - in extended package. In the context of a process that is synchronized by a clock signal, in a behavioral description, signals implicated in simple/ multiple signal assignment generate memory during synthesis.

An analogous rule can be formulated for variables: Inside a process, a variable that must hold values between iterations of the process implies memory elements. A variable that is set but not used between synchronization statements infers memory; a variable that is read before being assigned also infers memory.

The context is not restrictive, as all concurrent statements are equivalent to processes (excepting direct instantiation). For called subprograms the rules of memory inference can be deduced directly: pure functions (no side effects) do infer memory - while procedures (side effects) do not. All types of hierarchies are implied in this approach: design abstraction levels, block structure, class framework, symbolization and knowledge hierarchies.

We combined Partial Scan methods to optimize the order to add memory elements to the scan chain, at behavioral level. An adaptable interface assures the translation, in both senses, from behavioral hard/ soft description to a structural representation of the required behavior. The partial-scan selection uses a knowledge base to generate the weighted directed graph (flip-flops, combinational paths) and to return to text the differences caused by transformation for testability improvement.

The rules of correspondence between description object (signal/ variable) assignments and registers, and those to translate the data flow in the behavioral specification to weighted arcs in the graph counterpart and to combine different testability measures in node weights, guide the first step. The second step is solved by incrementing rules for the hardware-software description.

Partial-scan needs for the return translation a pointing scheme for the scanned objects among signals/ variables of the behavioral specification. This is managed by an adequate data structure in HDL. In principle, flip-flops are selected for scan, but when a register is used in parallel, it is candidate entirely for scan. The variables/ signals inferring memory are testability-related sorted to select incrementally the scan elements that will be eventually mapped to the scan register.

HIERARCHICAL INTELLIGENT SIMULATION

Essential relations are extracted from non-formalized research on the human mind before searching conscience models enabling intelligent simulation.

Faith and Intelligence are ☯ in our life (Way, Truth, Life)
Faith = (Inspiration, Intuition, Imagination)
Intelligence = (Consciousness, Adaptability, Intention)
Conscience = (Consciousness, Inspiration)

Conscience builds the non-deterministic interface between unconscious faith and the conscious intelligence.
// Concept = (interface, kernel, messenger).

Human = human (Humanity);
human ∈ Faith × Intelligence → Faith × Intelligence;
Humanity = (humans Set, evolution-oriented Structure).
evolution ∈ (Hunger, Fear, Love)
× (Technology, Science, Art)
→ (Technology, Science, Art).
Mathematics ⊂ Art = Human :: beauty-oriented
activity (Science, Technology).
Physics = (natural ∪ social) Science
= Human :: truth-oriented activity (Art, Technology).
Technology = Human :: good-oriented
activity (Art, Science).
Simulation∈ Behavior × Structure ⇐ Knowledge;
Knowledge ⇐ Intelligence :: information();
Imagination ⇐ | Intuition - Consciousness |;
Intention ⇐ | Inspiration - Adaptability |;
Adaptability ⇐ simplifying_Abstraction (Imagination);
Consciousness ⇐ reflexive_Abstraction (Intention);

The relations defining informally the class Human are oversimplified in order to move towards intelligent simulation. Although we claim they are intuitive and hope they are inspired, to begin, we neglect the essential but too primitive to understand intuition and inspiration, so formalizing the simplifying abstractions mainly by the simulation hierarchy type, and the reflexive abstraction by the knowledge hierarchy type:

Conscience = knowledge (simulation (Conscience))

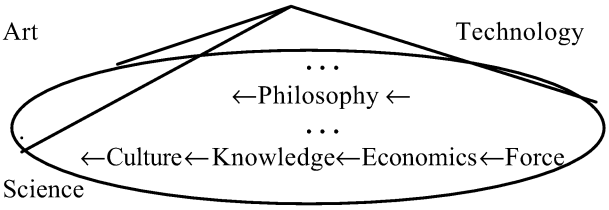
The fixed-point relation suggests to model conscience by association of a knowledge level to any hierarchical level of the simulation process. To solve the problem we build a metric space where knowledge°construction is a contraction - the elements implied in the construction get closer to one another in the formal understanding of the formal construct. Considering the general relations between the parts of the faith-assisted intelligence:

Conscience = knowledge (intention (Inspiration, simulation (imagination (Intuition, Conscience))))

Mathematics contains structures that suggest to be used for self-referent models. The richest domain in this sense is functional analysis, which integrates algebra, topology and order (Kolmogorov 1977):

- contractions and fixed points in metric spaces,
- reflexive normed vector spaces,
- inductive limits of locally convex spaces,
- self-adjoint operators of Hilbert spaces,
- inversable operators in Banach algebra.

Evolution is linked to the initial design of mental faculties for surviving of the whole system, but also to the space-time context for communication between intelligent agents. Conscience is self-awareness of individual faith and intelligence, as well as of the relation to the local context (society) and to the global one (Universe/ Reality). To appear it needed self-knowledge, what could have resulted from community conscience featured by an eternal human structure, e.g., from the past: shepherds, farmers, sailors, Africans, Amerindians, ... Each individual recognized himself in his cohabitants, being most adaptable and having a lot of intuition. The common measure evolution



implies the construction of correspondingly intelligent agents to manage the lower stages and to concentrate on the higher ones. E.g., industry enabled the mechanization in agriculture preparing for the concentration on economics. Evolution is a multiple Divide et Impera et Intellige for conscience, generating the components lacking of the mind at start, then assisted by them:

- individual-social-universal conscience (subjective-contextual-objective) → *inspiration* ↓
- space-time (structure-behavior) → *imagination* ↓
- discrete-continuous (natural-real) → *intention* ↓
- beauty-truth-good (art-science-technology).

The convergence process of evolution demands struggle against time, with structure as ally. Structure is sometimes too conservative, so it has to be reconfigured, at abstract levels, e.g., a plan, as at concrete ones. Conscience needs continuous feedback, not only discrete recurrence. Social and individual conscience are mostly divergent nowadays, i.e., we only performed Divide et Impera, neglecting *et* Intellige. It's high time to correct this! Intelligence in evolution is the faculty to transform (analyze/ synthesize/ modify) abstract/ natural/ artificial objects, and representations, in the correspondent worlds of arts, sciences and technologies, especially hierarchical reflexive:

ideas about ideas, how to get to ideas, objects to transform objects, representations on representations, how to build/ understand representations.

Recurrence is confined to discrete worlds, while abstraction is not. This difference suggests searching for understanding based on mathematical structures that order algebra into topology. Recurrence of structures and operations enables approximate self-knowledge, with improved precision on higher levels of knowledge hierarchies. A continuous model for hierarchy levels, without loosing the hierarchy attributes, would offer a better model for conscience and intelligence. A possible interpretation of knowledge hierarchies is: real time of the bottom levels (corresponding to primary knowledge/ behavior/ methods) is managed at upper levels (corresponding to concrete types/ strategies/ models) and abstracted on highest levels (corresponding to abstract types/ theories/ techniques).

A topology on the space of symbolic objects permits grouping items with common properties in classes. A dynamically object-oriented internal representation results, that can be adapted to the different hierarchy types. Topological concepts, as neighborhood, or concepts integrating mathematical structures, as closure, can be applied in verification and optimization, for objects and classes.

The simulation environment prepares a framework for representing entities and relations of the system to be simulated as general knowledge about the simulated universe. Knowledge-hierarchy-oriented architecture, both at environment and simulation component level, ensures flexibility of the framework realization, by defining it precisely only in the neighborhood of solved cases. For representation, this principle offers the advantage of open modeling. The user describes models, following a general accepted paradigm that ensures syntactic correctness, leaving the meaning to be specified by user-defined semantic functions that control the simulation.

E.g., a module in an unfinished design can be characterized by constraints regarding its interaction to other modules; the constraints system is a model, open to be interpreted, thus implemented, differently, adapting to criteria in a non-monotonic logic.

Let $(U, \{H_i \in S_h\})$ be a universe structured by different hierarchies H_i and S_h the set of hierarchies defined on it.

$$H = (Rel_{eq}, \{(Level_i, Structure_j) \mid j \in S_l\}, Rel_{ord}, \{A_j \mid j \in S_l\})$$

H is a generic hierarchy, S_l the set of hierarchy levels, Rel_{eq} the equivalence relation generating the levels, $Structure_j$ the structure of level j, Rel_{ord} the (total) order relation defined on the set of hierarchy levels, and $A_j \subset Level_{j-1} \times Level_j$, $j \in S_l$ the abstraction relation. U is a category, e.g., containing Hilbert spaces with almost everywhere-continuous functions as morphisms, enabling different ways to simulate self-awareness.

A hierarchical formal system can be defined. Considering self-adjoint operators as higher-level objects of the knowledge hierarchy, these levels can approach self-knowledge in the context of knowledge about the inferior levels as of the current one, and having some qualitative knowing about the superior levels. The self-knowledge raises as the abstraction corresponding to the hierarchy level.

1. $(U, \{H_i \in S_h\})$, $card(U) > \aleph_0$ // hierarchical universe
2. $\Sigma = F \cup L \cup A \cup K$ // functional objects
   $F = \{f \mid f \in U^* \to U\}$ //global functions
   $L = \{f \mid f \in Level_j^* \to Level_j\}$ //level structures
   $A = \{f \mid f \in Level_j^* \to Level_{j+1}\}$ //abstractions
   $K = \{f \mid f \in Level_j^* \times Level_{j+1} \to Level_{j+1}\}$ // knowledge abstractions
3. $I = \Sigma^* \cap R$ // initial functions
4. $R = \{r \mid r \in \Sigma^* \times R^* \to \Sigma \times R\}$ // transformation rules.

The correspondence problem, i.e., associating the knowledge hierarchy to the simulation hierarchy, is managed by natural transformations over the various functors of the different hierarchies regarding the simulated system. To complete the simulation of the intelligence's components, intention is first determined by human-system dialogue.

The alternative ways followed to extend the computability concept are suggested by approaches known from German literature, which is philosophy-oriented, trying to express essential ideas that link to the unconscious part of our mind. They respectively concentrate on the mental world of the good managed by technology, the physical world of the truth researched by science, and Plato's ideal world of abstractions discovered by arts.

1. Faust (Johann Wolfgang von Goethe): heuristics - risking competence for performance, basing on imagination, confined to the mental world.
2. Das Glasperlenspiel (Hermann Hesse): unlimited natural parallelism - remaining at countable physical suggestions, so in the Nature.
3. Der Zauberberg (Thomas Mann): hierarchical self-referential knowledge - needing to conciliate the discrete structure of hierarchy with the continuous reaction, hoping to open the way to Reality.

CONCLUSIONS

Conscience simulation demands transcending the present limits of computability, by an intensive effort on extensive research to integrate essential physical and mathematical knowledge guided by philosophical goals. Formalizing hierarchical descriptions, we create a theoretical kernel for self-organizing systems. A way to begin is hierarchical reconfigurable cosimulation. Applying Divide et Impera et Intellige to hierarchy types, using the formalism of categories, reveals their comprehensive constructive

importance based on structural approach, symbolic meaning, object-oriented representation. Further than modeling conscience to simulate intelligence we will be searching to comprehend inspiration, using Lebesgue measure on differentiable manifolds and non-separable Hilbert spaces. Perhaps even mathematics will have to develop more philosophy-oriented to approach intuition.

Simulability is computability by the power of continuum.

There are enough positive signs for this from analog electronics, control systems, mechatronics. Real progress towards this way of computation needs unrestricted mathematics, integrated physics and thinking by analogies. Evolution implies the separation of faith and intelligence, so we have to better understand both, integrating them to human wisdom, to be divided further to get more human. Metaphorically phrased, our searches and researches should have as axioms:

God is unique.
Uncountable are His ways.
Hierarchical are His plans, so we hope.

REFERENCES

Ageron, P. 2001. "Limites inductives point par point dans les categories accessibles". *Theory and Applications of Categories* 7, No. 1, 313-323.

Blum, L.; F. Cucker; M. Shub; and S. Smale. 1998. *Complexity and Real Computation*, Springer, Heidelberg/ NewYork.

Bourbaki, N. 1966. *Éléments de Mathématique. Théorie des Ensembles. Structures*. Hermann, Paris.

Ciaffaglione, A. and P. diGianantonio. 2001. "Constructive Real Numbers". *Proceedings of TYPES. Lecture Notes in Computer Science* 1961, 2277-2292.

Goldstein, S.C. and M. Budiu. 2001. "Nanofabrics: Spatial Computing using Molecular Electronics". *Proceedings of the 28th Annual Symposium on Computer Architecture* (Göteborg, Sweden, Jun.17-19). IEEE, Piscataway, NJ. , 178-191.

Hofstader, D. 1979. *Gödel, Escher, Bach: an Eternal Golden Braid*. Basic Books Inc., New York.

Kolmogorov, A. and S. Fomin. 1977. *Analyse Fonctionnelle.* MIR, Moskow.

Miller, R.; V.K. Prasanna-Kumar; D. Reisis; and Q.Stout. 1993. "Parallel Computations on Reconfigurable Meshes." *IEEE Transactions on Computers* 42, No.6, 678-692.

Niculiu, T. and S.D. Cotofana. 2004. "Hierarchical Simulation of Intelligence", *Proceedings of the 16th Conference on Systems Research, Informatics and Cybernetics* (Baden-Baden, Germany, Jul.29-Aug.5). The International Institute for Advanced Studies on Systems Research and Cybernetics, Windsor, Canada, keynote, 45-51.

Niculiu, T. and S.D. Cotofana. 2003. "Hierarchical Reconfigurable Simulated Intelligence Templates", In *Proceedings of the International Conference on Intelligent Systems and Control*. (Salzburg, Austria, Jun. 22-25). The International Association of Science Technology and Education for Development, Calgary, Canada , 39-44.

Niculiu, T.; C. Aktouf; and S.D. Cotofana. 2002. "High-Level Intelligence oriented Simulation". In *Proceedings of the International Semiconductor Conference* (Sinaia, Romania, Oct.8-10). IEEE, Piscataway, NJ. , 381-385.

Penrose, R. 1989. *The Emperor's New Mind*. Oxford University Press, Oxford.

Plesu, A. 2003. *About angels*. Humanitas, Bucharest,

Rabaey, J. 1997. "Reconfigurable Computing". In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing* (München, Germany, Sep.22-24). IEEE, Piscataway, NJ. , 127-132.

Traub, J.F. 1999. "A Continuous Model of Computation". *Physics Today*.5, No.5, 39-43.

Zeigler, B.; H. Praehofer; and T. Kim. 2000. *Theory of Modeling and Simulation*. Academic Press, Oxford.

Zhong, N. and K. Weihrauch. 2003. "Computability Theory of Generalized Functions". *Journal of Automated Computer Machinery.*50, No. 6, 574-583.

BIOGRAPHIES

TUDOR NICULIU is Professor at the Electronics, Telecommunications and Information Technology Faculty of the University POLITEHNICA in Bucharest and Senior Researcher at the Center for New Electronic Architectures of the Romanian Academy. He is looking for hierarchical integration of different domains, as: hardware-software, discrete-continuous, electrical-non-electrical, in order to simulate intelligence to understand it and to apply it to intelligent simulation. Therefore he formalizes hierarchies. He wants to contribute at comprehension and surpassing of the computability limits, functionally or structurally. Since 19991 he teaches and researches at the same institution (PhD 1995, MS 1985). Before he was Senior Researcher at the R&D Institute for Electronic Components in Bucharest, researching and designing hierarchical simulation of integrated circuits. He also studied Mathematics at the University of Bucharest (MA 1994).

ANTON MANOLESCU is Professor at the Electronics, Telecommunications and Information Technology Faculty of the University POLITEHNICA in Bucharest. He researches, teaches and masters Phd students on Analog Integrated Circuits and Technologies. He published more than 100 articles in international journals and conference proceedings. He has 3 invention patents and a prize of the Romanian Academy for his outstanding activity.

SORIN COTOFANA is Professor at the EE Dept., Delft University of Technology. He received the MS degree in Computer Science from the POLITEHNICA University of Bucharest, Romania, and the PhD degree in Electrical Engineering from Delft University of Technology, The Netherlands. He worked for a decade with the Research & Development Institute for Electronic Components in Bucharest, working on structured design of digital systems, design rule checking of integrated circuits' layout, logic and mixed-mode simulation of electronic circuits, testability analysis and image processing. His research interests include: computer arithmetic, custom computing machines, embedded systems, parallel architectures, logic design, computational geometry, neural networks, and CAD.

TUTORIAL

GRID COMPUTING AND STOCHASTIC DISTRIBUTED SIMULATION

David R.C. Hill
ISIMA / LIMOS UMR CNRS 6158
Blaise Pascal University
BP 10125 – 63177 Aubiere Cedex France
www.isima.fr/hill

ABSTRACT

Grid computing, can be considered as distributed computing taken to the next evolutionary stage. This paper presents an historical perspective of distributed computing followed by the concepts of grid computing and a presentation of some grid systems. This paper also deals with the distribution of stochastic simulations and experiments (in particular Monte Carlo simulations). The different techniques in use to distribute random numbers are also presented as well as future research directions.

INTRODUCTION

Before getting to the current status of grid computing, lets start with an historical perspective of distributed computing. If we had to trace what could be the beginning of distributed systems, we have to recall to our mind that at the beginning of the sixties, Leonard Kleinrock a scientist working at MIT, published a paper on packet switching entitled "Information flow in large communication Nets" (Kleinrock 1961). Still at MIT, J.C.R. Licklider envisioned a global interconnected set of computers where everyone could access programs and information from any sites. The first book on the packet switching theory was published in 1964, again by Kleinrock (Kleinrock 1964). In 1965, the first wide area network was built with a low speed telephone line connecting Massachusetts and California time-shared computers that were able to work together. Also in 1965, the US Department of Defence Advanced Research Project Association (DARPA), started to work on ARPANET (the Advanced Research Program Agency Network). In 1966, the concept of "computer network" was introduced at DARPA and published in 1967. In 1969, the File Transfer Protocol (FTP) helped in distributing the RFCs (Request For Comments) which were fast and easy ways to share ideas amongst network researchers. It was also at the end of the sixties that Ken Thompson, Dennis Ritchie and others started to work on what will become Unix at AT&T Bell Laboratories. At the same time in 1969, ARPANET was connecting 4 universities & research institutes (Stanford Research Institute, UCLA, UC Santa Barbara and the University of Utah). In 1972, the InterNetworking Working Group (INWG) was built to control the growing network and the first e-mails were introduced to public. The next year saw the introduction of the Ethernet Protocol at Xerox PARC (Palo Alto Research Center). Year 1974 saw both the arrival of Telnet, a commercial version of ARPANET and the rewriting of Unix V.4 with the C language, thus enabling a better portability of this operating system among different computers. In 1979, the USENET newsgroup was introduced by students from North Carolina and Duke Universities. The beginning of the eighties saw the spreading of IBM personal computers and its compatibles. The term Internet was introduced in 1982, and in 1983, the old NCP (Network Control Protocol) introduced in 1970 was replaced by the current TCP/IP protocol. Five years later, in 1988, the spreading of the Internet Worm showed the vulnerabilities of the Internet, thus leading to the formation of the Computer Emergency Response Team (CERT) to address security issues. At the Palo Alto DEC Research Center, the distribution of computing tasks sent via e-mails to workstations inside and outside their laboratory was successfully achieved in 1988 (results were also collected via e-mails).

A few years later, in 1991, the concept of World Wide Web is born and in the same year Linus Torvalds distributed the source code of the Linux kernel as freeware. In 1993, the first Web browser, named Mosaic is released applying the Web concepts developed at CERN ; in the same year, the first RSA Security challenge is won by a team of 600 volunteers providing their distributed computing resources. In 1994, version 1.0 of Linux, produced by Linus Torvalds and a world wide team of hackers, is available for download. The annual growth rate of the Web between 1993 and 1994 is over 300%. The following year saw the introduction of an alpha version of Java, an object-oriented language with a portable byte-code able to be transmitted and executed on the Internet. The GIMPS distributed computing project, searching for larger and larger prime numbers is launched in 1996. On August the 26th of 1997, the PI challenge is launched by Fabrice Bellard to find the 1000th billion binary digit of PI, this goal is reached on September 22nd. In 1999, the concept of Internet computing was providing at least 3 times more computing power (33 Teraops) than any other supercomputers with the Seti@home project of Berkeley University, which analyzed data from the famous Arecibo radio-telescope. Clients software were available for Windows, Macintosh-OS and Unix. More useful projects appeared with the new millennium. In year 2000, the Casino-21 project goal was to simulate the dynamics of the earth's climate for the next century. In year 2001, Barabasi from Notre Dame University introduced in (Barabasi et al. 2001) the concept of parasitic computing, explaining and showing how Internet protocols could be used to compute complex linear programming problems. At the end of year 2001, the French Decrypton project was setup to compute and

compare 50000 protein sequences with a distributed client software developed by Genomining and IBM. More than 75000 Internet users participated to reach the first results in 2 months, 1170 years would have been necessary to achieve the same result on a single computer. The collected data is now publicly available for the international community of scientists working in bioinformatics. A taxonomy of distributed technologies that have emerged following all these events can be found in a paper written with Fabrice Bernadi (Bernardi et al. 2004). The former includes classical client/server systems, metacomputing, massive distributed architectures (P2P, Internet computing,…) agent-based systems and the communication models able to be implemented in distributed systems. In addition, many presentations and definitions of distributed systems can be found in the literature from the Flynn classification (Flynn 1972) to various recent books from Andrew Tannebaum. In this paper the main focus will be given to grid computing concepts and systems.

CONCEPTS AND SYSTEMS OF GRID COMPUTING

Grid computing is defined as "coordinated resource sharing and problem solving in large, multi-institutional virtual organization" by (Baker et al. 2002). The term Grid suggests a computing paradigm similar to an electric power grid. This was a very new concept and is currently under a heavy phase of research and development. A quite complete review can be found in (Allan and Ashworth 2001). (Buyya 2002) provides an economical perspective of this kind of systems, while (Paindaveine 2003) provides a review of Grids in the health sector. Grid applications couple resources that cannot be replicated at a single site or may be globally located for other reasons. Thus, a Grid is an infrastructure that can unify globally and diverse resources. The main issues for computational grids are heterogeneity since a grid involves a multiplicity of heterogeneous resources; scalability since a grid might grow from few resources to millions; dynamicity or adaptability since the probability of some resource failing is high with so many resources. There are four components necessary to form a grid :

- a Grid Fabric comprising all the resources (computers, clusters,...) geographically distributed and accessible from anywhere on the Internet;
- a Grid Middleware offering core services such as remote process management, co-allocation of resources, storage access,...;
- a Grid Development Environment offering high level services that allows programmers to develop applications and brokers that act across global resources;
- some Grid Applications and Portals developed using Grid-enabling languages.

There are many grid projects worldwide. Among the Grid Computing Systems we try to select the most recognized one, which are however still in their research and development phases. Some of them, like Globus, have been employed in very strong and precise researches (see references). More grid projects can be found at (http://www.globus.org/about/related.html), at (http://www.gridcomputing.com) or at (http://dsonline.computer.org/gc/gcprojects-european.htm), and many of the systems presented here are described more precisely in the excellent review provided by (Kacsuk and Vajda 1999). Here is a non exhaustive list, excerpt of grid computing systems we selected in (Bernardi et al 2004) :

- Globus: Globus is the most known and used of grid-enabling systems. It is released as a toolkit proposing a set of components that can be used independently or together. We can cite as components a resource allocation manager, security infrastructure, a monitoring and discovery service or a failure detector as described in (Foster and Kesselman 2001) or (Allcock et al. 2001). Globus 3.0 is the first full major implementation of an open standard for grid computing OGSA (Open Grid Services Architecture) well described at http://www-106.ibm.com/developerworks/grid/library/gr-visual/index.htm%l) and presented in (Alpdemir et al. 2003). Main URL: http://www.globus.org
- ALiCE: ALice (Adaptive and scalable Internet-based Computing Engine) is a platform-independent, Java based grid middleware. It comprises three types of entities: a resource broker, service producers and service consumers. Main URL: http://www.comp.nus.edu.sg/~teoym/alice/alice-tech.htm
- Condor: Condor is defined as a specialized workload management system for computer-intensive jobs. The goal of the project is to develop mechanisms and policies that support high-throughput computing on large collections of computing resources. (Thain et al. 2003), (Basney et al. 1997) and (Tannenbaum et al. 2002) present this project. Main URL: http://www.cs.wisc.edu/condor/
- Grid Engine: Sun ONE Grid Engine is available in two versions: SGE and SGE Enterprise Edition. This middleware requires no alterations to applications to be distributed. More information can be found in (Coomer and Chaubal 2002). Main URL: http://wwws.sun.com/software/gridware/
- GriPhyN: GriPhyN is a collaboration of computer science researchers and experimental physicists to enable a grid system following two approaches: apply a methodical and organized data management using the virtual data concept and developing automated request scheduling mechanisms in order to make large grids easy to use as single computers. Main URL: http://www.griphyn.org/index.php
- Legion: Legion enables the connection of networks of computers using an independent element approach. There is no central element and the proposed framework permits scheduling and distributing processes as in a single, virtual machine. More information can be found in (Natrajan et al. 2001). Main URL: http://www.cs.virginia.edu/~legion/

- Metacomputer OnLine: Metacomputer OnLine (known as MOL) is designed as an open, extensible software system comprising a variety of software modules. In contrast to other grid environments, MOL is not based on specific models or tools. Main URL: http://www.uni-paderborn.de/pc2/projects/
- NetSolve: NetSolve is a grid framework focusing on three main points: ease of-use for the end-user, efficient use of the resources and the ability to integrate any arbitrary software component as a resource into the Net-Solve system. More information, as well as a grid concept description can be found in (Arnold et al. 2001). Main URL: http://icl.cs.utk.edu/netsolve/

STOCHASTIC PARALELL SIMULATION

Parallel and distributed simulation has been extensively studied for deterministic models. The major references in this field (Chandy and Misra 1979) for conservative approaches (Jefferson and Sowizral 1985) for Time Warp an optimistic approaches ; (Fujimoto 2000) presents the current state of the art in this research direction. However, research on stochastic parallel simulation is much tougher and this domain is less studied. Few are aware of the potential dangers associated to the distribution of sequential stochastic simulations. A well known paper in this domain warned scientists at the end of the last century to be very careful with parallel Monte Carlo (Hellekalek 1998). Indeed, if sequential stochastic simulations always require a statistically sound RNG (Random Number Generator), in the case of parallel and distributed stochastic simulations, this requirement is crucial. Major scientists in this research field such as Paul Coddington state that: *"Random number generators, particularly for parallel computers, should not be trusted. It is strongly recommended that all simulations be done with two or more different generators, and the results compared to check whether the random number generator is introducing a bias"* (Coddington 1996). The main advice that specialists recommend is that random numbers should be generated in parallel, i.e. each Logical Processor (LP) should autonomously get its own sub-sequence of a global random sequence. Designers of parallel stochastic simulations always have to answer this fundamental question: how can we make a safe RNG repartition in order to keep, on the one hand, efficiency, and on the other hand a sound-statistical quality of the simulation in order to obtain credible results ? Indeed the validation of such parallel simulations is a critical issue. Many research studies have been undertaken to design good sequential RNGs. Many others concern Parallel Random Number Generators (PRNGs). It appears that the key focus, which is how to assess the quality of random streams, remains a hard problem, and many widely-used techniques have been shown to be inadequate for some specific applications.

An obvious way to get parallel random numbers streams is to partition a sequence of a given generator into suitable independent sub-sequences. This can be done in three major ways (James 1990) (Traore and Hill 2001) :

1. The Leap Frog (LF) method allocates in turn the random numbers of a sequence to a partition of streams like a deck of cards dealt to card players. Thus, for a partitioning of a global sequence $\{x_i, i = 0,1,2,\ldots\}$ into N parallel streams, the j^{th} stream is $\{x_{kN+j-1}, k = 0,1,2,\ldots..\}$. Given the period p of the global sequence, the period of each stream can reach p/N.
2. The Sequence Splitting (SS) method splits a sequence into non overlapping contiguous blocks. Thus, for a partitioning of a sequence $\{x_i, i= 0, 1, 2, \ldots\}$ into N streams, the j^{th} stream is $\{x_{k+(j-1)m}, k = 0, \ldots, m-1\}$, where m is the length of each stream. One major difficulty is the determination of a good value for 'm', which must be chosen so that each stream is long enough to achieve the stochastic simulation performed by the corresponding processes.
3. The Independent Sequences (IS) method builds a partition of N streams by initializing the same generator with N different seeds. Such a technique can lead to overlapping streams, since a random number generated in one stream can match the seed used for another stream. Some generators exhibit several sub-cycles that can be encapsulated into other streams. Such generators seem to be well adapted to the IS method, since the seeds to use for parallel streams can be the first terms of the sub-cycles.

Each way has its own specific flaw and whatever the solution retained, we have to check the correlations between subsequences. Indeed, the correlation phenomena observed in a sequential RNG are often induced in the parallel streams that are obtained from this generator. For instance, an initial correlation between distant numbers of a sequence can lead to auto-correlation inside parallel streams produced with the LF method, or cross-correlation between parallel streams produced with the SS method. Variants have been developed for three methods presented above such as the shuffling leap frog or the parameterization method which is a variant of the IS technique (See the work of Prof. Mascagni Team http://www.ncsa.uiuc.edu/Apps/SPRNG/www/paper/). Its principle is to parameterize both the seed and recursive function. One of the main contributions of this variant is that it results in a scalable period, i.e. the number of different random numbers that can be used increases with the number of parallel streams. An interesting parameterization method is used in (Percus and Kalos 1989) to get parallel streams from an LCG, by choosing for the j^{th} stream a multiplier a(j) and an additive constant c(j). It is shown in this case that good results can be obtained if c(j) is the j^{th} prime number $\leq \sqrt{(m/2)}$ where m is the modulus. However testing pseudo-random numbers in their parallel context is essential. The usual approach in testing a PRNG is to apply standard tests which are available for sequential RNG to each random number parallel streams, and then to all streams combined (Mascagni et al. 1995). New techniques appear which are specific to PRNGs

http://archive.ncsa.uiuc.edu/Science/
CMP/RNG/www/paper/index.html). A parallel version of the well known spectral test has been developed in (Percus and Kalos 1989) for parallel LCG. So far, little work has been done in this area. Since not many rigorous results are known about the properties of PRNG, more stringent and varied tests are necessary. The next section will present an application were we have chosen the Sequence Splitting parallelization method.

SAMPLE APPLICATION AND DISCUSSION

The use of Monte Carlo Simulations for clinical treatment is now showing its efficiency to obtain better precision in cancer treatments with radiotherapy than what is currently obtained with analytical methods. However, the computing time using Monte Carlo calculations must stay comparable to what is currently obtained with analytical calculations. Since particle emission can be simulated independently, the parallelization of Monte Carlo simulations by splitting pseudorandom numbers generator sequences into a multitude of streams is a simple, rigorous and generic method (i.e. it can be applied for each Monte Carlo codes used in physics with a pseudorandom number generator). In (Maigne et al 2004) we have recently presented the comparison between a local and a parallel computation of some physical results of interest showing that the output data stayed unchanged and therefore validated the parallelization method used. Additional tests should be conduced with different pseudorandom numbers.

Despite many constraints due to Network latency and Grid load, the gain in computing time obtained by splitting the simulations were very encouraging. To speed up the calculations, we can think about enabling a new submission for the jobs that are waiting for a too long time in the Grid batch queue in order to be computed. In the future, it is envisaged to setup a waiting time in the batch queue equivalent to 10 % of the total estimated computing time of the jobs, after this delay, jobs still in the waiting status will be stopped and submitted again. The new submission will be achieved on another grid site where either the number of free CPUs is consequent or the running status is reached quickly for the submited jobs. Additional optimization can also be done for Monte Carlo simulations using unrolling techniques (Hill 2003). Local files of pre-computed random number subsequences can be accessed with memory mapping techniques on local worknodes without efficiency losses compared to a classical memory access. Indeed the use of random numbers in a Monte Carlo simulation is always predictable and cacheable in processor caches if pre-computed and stored in arrays.

CONCLUSION

Grid systems are very promising for Monte Carlo Simulations. Parallel Monte Carlo simulation can be complex to setup with sound statistical bases, however parallelizing techniques for pseudo-random numbers are now carefully studied. Recent work precisely analyzed Large-Scale Grid-Based Monte Carlo Applications (Li and Mascagni 2004). Independent replications should now always be done in parallel, this will always imply a performance gain (Li and Mascagni 2003), boostraping can also benefit of this new computing power. The current perspectives should now focus again on basic operation research and performance evaluation studies that should be done to optimize job submission with tools like QNAP2 (Queuing Network Package 2) or with dedicated software like GridSim. The GridSim toolkit is an Australia based grid research project that enables users to conduct design and performance evaluations of scheduling algorithms. The primary objective of such tools is to is to create simulated grid environments that reflect real-world scenarios involving a varying number of resources and users with different (policy) requirements (Buyya 2002). Additional simulations, corresponding to metamodels have to be used to test resource allocation techniques and the effectiveness of resource brokers and scheduling algorithms. Current grid systems are in development phase and there are limits to how many machines can run grid jobs simultaneously. When a user is lucky, he may get 20 machines, but the design of large parallel applications where thousands of machines with many different load conditions, many different kinds of user scenarios should be considered for the evaluation of Grid scheduling systems. From this paper, we hope it is understood that Grid computing is not a silver bullet enabling any application to run faster and faster. There are many applications that cannot be parallelized, many others will need a large amount of work to allow better performances and in addition the grid configuration, reliability and infrastructure will also play an important role in the general performance.

REFERENCES

(Allan and Ashworth 2001) R.J. Allan and M. Ashworth. A Survey of Distributed Computing, Computational Grid, MetaComputing and Network Information Tools. Technical report, UKHEC, 2001.

(Allcock et al. 2001) W. Allcock, A. Chervenak, I. Foster, L. Pearlman, V. Welch, and M. Wilde. Globus Toolkit Support for Distributed Data-Intensive Science. In Proceedings of Computing in High Energy Physics (CHEP '01), 2001.

(Alpdemir et al. 2003) M.N. Alpdemir, A. Mukherjee, N.W. Paton, P.Watson, A.A. Fernandes, and A. Gounaris. OGSA-DQP - A Service-based Distributed Query Processor for the Grid. In EPSRC, editor, Proceedings of UK e-Science All Hands Meeting Nottingham, 2003.

(Arnold et al. 2001) D.C. Arnold, S.S. Vahdiyar, and J.J. Dongarra. On the Convergence of Computational and Data Grids. Parallel Processing Letters, Vol. 11, N°2&3, pp. 187-202, 2001.

(Baker et al. 2002) M. Baker, R. Buyya, and D. Laforenza. Grids and Grid Technologies for Wide-Area Distributed Computing. Software - Pratice and Experience, Vol. 488, 2002.

(Barabasi et al. 2001) A.L. Barabasi, V.W. Freeh, H. Jeong, and J.B. Brockman. Parasitic Computing. Nature, Vol. 412, N°30 August 2001.

(Basney et al. 1997) J. Basney, M. Livny, and T. Tannenbaum. High Throughput Computing with Condor. HPCU News, Vol. 1, N°2, 1997.

(Bernardi 2004) F. Bernardi, J-F. Santucci and D. Hill. A Survey of Distributed Systems in Bioinformatics, LIMOS Research Report 04-02. 2004.

(Buyya 2002) R. Buyya. Economic-based Distributed Resource Management and Scheduling for Grid Computing. PhD thesis, Monash University, 2002. Melbourne, Australia.

(Chandy and Misra 1979) K.M.Chandy and J. Misra, Distributed Simulation. IEEE Transactions on Software Engineering, Vol. 5, N°55, pp. 440-452. September 1979.

(Coddington 1996) P. Coddington, "Random Number Generators For Parallel Computers", *NHSE Review*, Second Issue, 1996.

(Coomer and Chaubal 2002) J. Coomer and C. Chaubal. Introduction to the Cluster Grid - Part 1 and Part 2. Technical report, Sun BluePrints, 2002.

(Flynn 1972) M.J. Flynn. Some Computer Organizations and their Effectiveness. IEEE Transactions on Computers, C-21, 1972.

(Foster and Kesselman 2001) I. Foster and C. Kesselman. Globus - A Metacomputing Infrastructure Toolkit. International Journal of Supercomputer Applications, Vol. 11, N°2, 2001.

(Fujimoto 2000) Fujimoto R.M. 2000. Parallel and Distributed Simulation Systems. Wiley Interscience January. 2000.

(Hellekalek 1998) P. Hellekalek, "Don't trust parallel Monte Carlo!", *Proceedings of the 12th Workshop on PADS*, Banff, Alberta, Canada, pp. 82-89, 1998.

(Hill 2003) Hill D. 2003. URNG: A portable optimisation technique for software application requiring pseudorandom numbers. *Simulation Modelling Practice and Theory*. Vol. 11, pp. 643-654.

(James 1990) James, A Review of Pseudorandom Number Generators, Computer Phys. Comm. Vol. 60, pp. 329-344, 1990.

(Jefferson and Sowizral 1985) Jefferson D. Sowizral H., "Fast concurrent simulation using the time warp mechanism". Distributed Simulation SCS, Vol 15 n° 2. January. San Diego. pp. 63-69. 1985.

(Kacsuk and Vajda 1999) P. Kacsuk and F. Vajda. Network-Based Distributed Computing - Metacomputing. Technical report, ERCIM, 1999.

(Kleinrock 1961) L. Kleinrock. Information Flow in Large Communication Networks. RLE. Quarterly Progress Report, July 1961.

(Kleinrock 1964) L. Kleinrock. Communication Nets - Stochastic Message Flow and Delay. McGraw-Hill, 1964. later re-issued by Dover Books.

(Li and Mascagni 2003) Y. Li and M. Mascagni, Improving Performance via Computational Replication on a Large-Scale Computational Grid, Parallel Computing, Vol 29, pp. 69-94. 2003.

(Li and Mascagni 2004) Y. Li and M. Mascagni, Analysis of Large-Scale Grid-Based Monte Carlo Applications International Journal of High Performance Computing Applications (IJHPCA) Vol. 17 pp. 369-382. 2004.

(Maigne et al. 1994) L. Maigne, D. Hill, P. Calvat, V. Breton, R. Reuillon, Y. Legr and D. Donnarieix, Parallelization of Monte Carlo Simulations and Submission to a Grid Environment, Parallel Processing Letters, Vol. 14, N° 2, pp. 2004.

(Mascagni et al. 1995) M. Mascagni, S.A. Cuccaro and D.V. Pryor, Techniques for Testing the Quality of Parallel Pseudorandom Number Generators. *Proceedings of the 7th SIAM Conference on Parallel Processing for Scientific Computing*. pp. 279-284, 1995.

(Natrajan et al. 2001) A. Natrajan, M. Humphrey, and A. Grimshaw. Capacity and Capability Computing in Legion. In Proceedings of the 2001 International Conference on Computational Science, 2001.

(Paindaveine 2003) Y.P. Paindaveine. HealthGRID - Old Wine in New Bottle. In Healthgrid, Proceedings of the First European HealthGrid Conference, Lyon, France pp. 20-27, 2003.

(Percus and Kalos 1989) Percus O.E. and Kalos M.H., Random number generators for MIMD parallel processors. *Journal of Parallel and Distributed Computing*. Vol. 6. pp. 477-497, 1989.

(Tannenbaum et al. 2002) T. Tannenbaum, D. Wright, K. Miller, and M. Livny. Condor - A Distributed Job Scheduler, chapter 15. The MIT Press, 2002. Beowulf Cluster Computing with Linux.

(Thain et al 2003) D. Thain, T. Tannenbaum, and M. Livny. Condor and the Grid, chapter 11. John Wiley, Grid Computing. Making The Global Infrastructure a Reality edition, 2003.

(Traore and Hill 2001) Traore M., Hill D., "The use of random number generation for stochastic distributed simulation: application to ecological modeling". 13th European Simulation Symposium, Marseille, October 18th-20th, pp. 555-559, 2001.

ACKNOWLEDGEMENT

We give a special thank to Dr. Mamadou Kaba Traore, Dr. Fabrice Bernardi and to Lydia Maigne (Ph.D. student) for their help in the research presented in this paper.

SIMULATION METHODOLOGY

METHODOLOGY

KNOWLEDGE INTEGRATING IN SITUATIVE MODELLING SYSTEM FOR NATURE-TECHNICAL COMPLEXES

Alexander Fridman, Andrey Oleynik and Olga Fridman
Institute for Informatics and Mathematical Modelling of Technological Processes
Kola Science Center of RAS
24A Fersman str., 184209, Apatity Murmansk reg.,
Russia
E-mail: fridman@iimm.kolasc.net.ru

KEYWORDS
conceptual modelling, structured analysis and design, knowledge integrating.

ABSTRACT

The paper below introduces a situative modelling system (SMS) developed within the frames of Structured Analysis and Design Technology (Yourdon 1975 and 1989) and intended for state analysis and prognosis of hierarchical systems (Mesarovic et al. 1970) described according to General Systems Theory (Mesarovic and Takahara 1975). The distinguishing feature of the SMS is that it is to support an open domain model and integrate different forms of both data representation and expert knowledge. A conceptual model of the subject domain is used as a core of the SMS to maintain every stage of modelling. Combined processing of data coming from simulation modules of the object's structure components, embedded geographic information system and expert system constitutes the main difference between the model and prototypes.

INTRODUCTION TO SITUATIVE MODELLING

The term "Nature-Technical Complexes" (NTC) will here stand for territories, ecosystems, technologies, etc, functioning at an essential dependency on spatial features of their components and time. Experiments upon NTCs are usually impossible or prohibited because of either time lack, or high cost, or hazard of unrecoverable changes. That is why modelling is the main method of NTC behaviour' studying and forecasting. Due to unavoidable knowledge incompleteness regarding NTC, it is reasonable not to confine to classic analytic models only, but grant means for usage of experts' experience. Thus, an NTC model ought to be open for operative changes resulting from previous modelling stages. To unite all above-mentioned forms of knowledge representation into a modelling system, an integrated subject domain conceptual model (SDCM) was proposed (Fridman 1998). SDCM comprises elements of a subject domain and relations among them formalising object structures and cause-effect links, which are essential within frames of the current study of an NTC. At forming the NTC conceptual model, some alternate structures of the object under investigation are input into the model by using OR-relations in object decomposition and resource generation. A specific extension of the situative approach (Pospelov 1986) on the problem of NTC investigation based on the SDCM was proposed to compare alternative object structures (Fridman 1999). It is aimed at decision making support by choosing one of the possible structures to realize the object under investigation.

The above-mentioned openness of the SDCM leads to a necessity of model reconstruction and rules out any a priori calculation scheduling. Respectively, the modelling support system's software must offer means for prompt adjusting of software system composition and calculating process at a model change.

SDCM BASES

SDCM is a hierarchical conceptual model of an NTC. It is composed of three sets of elements, namely objects, processes and resources (data), which carry links and relations. The hierarchy of objects (components of the object under investigation) reflects their organizational relations. Every object may be assigned with a number of processes simulating transformation of a set of input resources into a set of output ones. To avoid problems with small changes of variables and provide common logic-computational data processing, resources in the SDCM are treated as discrete variables and respectively attributed with lists of values. This also allows detailing the link control up to each resource, thus raising description adequacy of links among object components and decision control at branching links among model elements, and providing processing of time data sets during simulation mode. As one more the SMS' distinct from prototypes, one may note that besides user-defined types of model elements, some automatically assigned categories of elements are brought into their description to provide more detailed formal analysis of the SDCM.

The set of names of resources (data) is split into the set of names of variables having numerical values (VAR category), and the set of names of parameters having string values (PAR category):

$$D::=<Var, Par>,\ Var::=\{var_i\},\ i=\overline{1,N_v};$$
$$Par::=\{par_j\},\ j=\overline{1,N_p}, \qquad (1)$$

where N_v and N_p stand for cardinality of the respective sets. Here and further symbols ::= denote equality by definition. If necessary, other symbols of BNF-notation are used as well. Data may constitute resources (quantitative characteristics) of objects and processes, then they have RES category. Variables may be also used as adjusting parameters of performance quality functions (criteria) for SDCM elements (ADJ category). Respectively, the set of names of variables can be split into the subset of names of resources and the subset of names of adjusting parameters:

$Var::=<Res, Adj>$ (2)

A separate GIS category comprises graphic characteristics of SDCM objects directly calculated by means of a geographical information system (GIS).

Objects in the SMS have three main attributes, namely a name, a user functional type determining the object structure and functions, and a name of the super-object dominating the given SDCM object (is not defined for the global object). Depending on their position in the object tree and on the map, three object categories are distinguished. They are primitive objects (LEAF category) having no details from the point of view of global modelling goal, elementary objects (GISC category) geographically linked to one GIS-element (either a polygon, or an arc, or a dot on a GIS-cover) and compound objects (COMP category) comprising several objects. The set of objects is defined as follows:

$$O = \{ {}_{\alpha} o^{\gamma}_{\beta_\alpha} \} ::= \bigcup_{\alpha=1}^{N_L} O_\alpha, \quad (3)$$

where: $\alpha = \overline{1, N_L}$ is the level number in the object tree the given object belongs to (L – the total number of decomposition levels);

$\beta_\alpha = \overline{1, N_\alpha}$ is the ordinal number of the object on its decomposition level;

$\gamma = \overline{1, N_{\alpha-1}}$ is the ordinal number of the super-object dominating the given object on the higher decomposition level;

O_α is the set of objects belonging to the level α..

To ensure the SDCM connectedness, one and only one super-object is supposed to dominate every object on the first decomposition level, that is:

$$O_1 ::= \{ {}_1 o^0 \beta_1 \}, \quad \beta_1 = \overline{1, N_1}. \quad (4)$$

The set of objects (4) can be split into subsets on categories of objects with no common elements:

$$O :: = O^{LEAF} \cup O^{COMP} \cup O^{GISC}. \quad (5)$$

Adding the set of elementary GIS-objects to the set (5), we will get the complete set of the SDCM objects:

$$O' ::= O \cup O^{ELEM}, \quad (6)$$

where the complete set of standard GIS-objects is as follows:

$$O^{GIS} ::= O^{GISC} \cup O^{ELEM}. \quad (7)$$

SDCM processes reflect data processing and may be realized differently depending on the assigned category. There may exist processes of three categories, namely internal processes (INNER category) having all input-output data within the same object, intra-level processes (INTRA category) that link pairs of SDCM objects having no mutual subordination, and inter-level processes (INTER category) describing data transfer along organizational links. The main attributes of processes are: a unique name, an executor's definition and a functional type determining the type of the data transformation for the process. The executor of a process determines its computer realization.

The scheme of the conceptual model in the SMS is as follows:

$$S_{SMS} ::= <O, P, D^{CM}, H, OP, PO, U>, \quad (8)$$

where: O is the set of the SDCM objects defined in (3);

$P ::= \{p_n\}$, $n = \overline{1, N_p}$ is the set of processes;

$D^{CM} \subseteq D$ is the set of the SDCM data (resources);

H is the object hierarchy relation, which takes the following form considering (3) и (4):

$$H = \bigcup_{\alpha=1}^{N_L - 1} H_\alpha, \quad (9)$$

where: $H_\alpha \subseteq O_{\alpha-1} \times B'(O_\alpha)$ are hierarchy relations for every level of the object tree and $B'(O_\alpha)$ means a partition of the set O_α;

$OP \subseteq O \times B(P)$ is the relation "an object – the processes generating its output data" and $B(P)$ means the power set (Boolean) of the set P;

$PO \subseteq P \times B(O)$ is the relation "a process – the objects generating its input data";

$U ::= U_p \cup U_o$ is the relation formalising the SDCM-based calculation process control. It has the following components:

$U_p \subseteq P \times B(Res)$ is the relation "a process – the control datum";

$U_o \subseteq O \times B(Res)$ is the relation "an object – the control datum".

Every process assigned to an object is described with the relation $OA \subseteq O \times B(P)$ "an object – the assigned processes". This relation is not included into the SDCM scheme since it is formed automatically unlike the relations H, OP и PO user-formed during the model creation. The relations defined within the model are convenient to represent as functions (10) partially defined upon sets O and P with ranges of values $B(P)$, $B(O)$, $B(Res)$ or $B'(O_\alpha)$. Names of the functions are marked with small letters used in the names of relations:

$h_\alpha : O_{\alpha-1} \to B'(O_\alpha)$, $(\forall o_j \in O_\alpha, \ \forall o_i \in O_{\alpha-1})$ $((o_j = h_\alpha(o_i)) \Leftrightarrow o_j H_\alpha o_i)$;
$op : O \to B(P)$, $(\forall o_i \in O, \ \forall p_j \in P)$ $((p_j = op(o_i)) \Leftrightarrow o_i OP p_j)$;
$po : P \to B(O)$, $(\forall o_i \in O, \ \forall p_j \in P)$ $((o_i = po(p_j)) \Leftrightarrow p_j OP o_i)$; (10)
$oa: O \to B(P)$, $(\forall o_i \in O, \ \forall p_j \in P)$ $((p_j = oa(o_i)) \Leftrightarrow o_i OA p_j)$;
$u_p: P \to B(Res)$, $(\forall p_i \in P, \ \forall res_j \in Res)$ $((res_j = u_p(p_i)) \Leftrightarrow p_i U_p res_j)$;
$u_o: O \to B(Res)$, $(\forall o_i \in O, \ \forall res_j \in Res)$ $((res_j = u_o(o_i)) \Leftrightarrow o_i U_o res_j)$

Ranges of values of functions (10) associated with cross-sections of the defined relations on an element of their domains of definitions are written in bold letters:

$\boldsymbol{h_\alpha}(o_i) ::= \{o_j: o_j = h_\alpha(o_i)\}$; $\boldsymbol{op}(o_i) ::= \{p_j: p_j = op(o_i)\}$;
$\boldsymbol{po}(p_j) ::= \{o_i: o_i = po(p_j)\}$; $\boldsymbol{oa}(o_i) ::= \{p_j: p_j = oa(o_i)\}$; (11)
$\boldsymbol{u_p}(p_i) ::= \{res_j: res_j = u_p(p_i)\}$; $\boldsymbol{u_o}(o_i) ::= \{res_j: res_j = u_p(o_i)\}$.

Cross-sections of the defined relations on a subset of their domains of definitions are defined as joins of all cross-sections on elements of this subset and denoted by analogy with (11). For instance, $\boldsymbol{h_\alpha}(O_i)$ where $O_i \subseteq O_{\alpha-1}$ means a set of objects from level α dominated with the subset of objects $o_j \in O_i$ belonging to the level α-1. Formulas below also use the total dominated subset of an object o_i :

$$\underline{\underline{\boldsymbol{h}}}^{\boldsymbol{\alpha}}(o_i) ::= \bigcup_{\alpha \le k \le L} \boldsymbol{h}^{\alpha}_{\boldsymbol{k}}(o_i) \quad (12)$$

The hierarchy relations H (8) allow to easily correlate any compound SDCM object with a subset of the set (7). The

subordination set $\underline{\underline{h}}^{\alpha}(o_i)$ of any object not belonging to the set of GIS-elements set (7) was proved to comprise the "chief subset" of the dominated GIS-elements which compose any alternative structure of this object, and a set of alternative GIS-elements revealing under a certain realization of the object:

$$(\forall o_i \in O \setminus O^{GIS})\ (\underline{\underline{h}}^{\alpha}(o_i) = O_0(o_i) \cup Alt(o_i)), \qquad (13)$$

where the symbol \ stands for the calculation of the set difference and

$$Alt(o_i) ::= \bigcup_{j=1}^{n} O_j(o_i), \qquad (14)$$

while $O_j(o_i) \subset O^{GIS}$, $j = \overline{0,n}$; $O_0(o_i) \cap O_j(o_i) = \varnothing$, $j = \overline{1,n}$.

The set O_0 comprises the GIS-elements that have no super-objects attributed by the OR-decomposition. The GIS representation of any alternative structure of the object o_i is realized by the set O_0 and one of the sets O_j:

$$Alt_j(o_i) ::= O_0(o_i) \cup O_j(o_i). \qquad (15)$$

We can claim that there are no alternative realizations for natural objects included in an NTC i.e.:

$$Alt(o_i) ::= \varnothing. \qquad (16)$$

The coordination (16) is considered as a limitation for admissible GIS-structures of natural objects unlike technical objects for which alternative structures are liable to comparative modelling.

Model of attributes in the SDCM is formed by the suite:

$$A_{SMS} ::= <\mathbf{N}, \mathbf{T}, \mathbf{n}, \mathbf{t}, \mathbf{E}, \mathbf{T}_e, t_p, t_r, T_h, t_{ho}>, \qquad (17)$$

where: $\mathbf{N} ::= <N_p, N_o, N_r>$ and $\mathbf{T} ::= <T_p, T_o, T_r>$ are sets of names and types for processes, objects and resources respectively;

$\mathbf{E} ::= <E_p, E_o, E_r>$ and $\mathbf{T}_e ::= <T_{ep}, T_{eo}, T_{er}>$ are sets of names and types for executors of processes, objects and resources respectively;

$\mathbf{n} ::= <n_p, n_o, n_r>$, $\mathbf{t} ::= < t_p, t_o, t_r>$ and $t_e ::= <t_{ep}, t_{eo}, t_{er}>$, where n_p: $P \to N_p$; n_o:$O \to N_o$; n_r: $D^{CM} \to N_r$; t_p: $P \to T_p$; t_o: $O \to T_o$; t_r: $D^{CM} \to T_r$; t_{ep}: $P \to T_{ep}$; t_{eo}: $O \to T_{eo}$; t_{er}: $D^{CM} \to T_{er}$ are functions describing the current set of elements;

$T_h ::= \{\&, \vee, *\} \cup N$ (N - the set of positive integers) is the set of admissible types of hierarchy relations among objects where classification is made by using the strictly exceptional OR. The application of different types of hierarchy relations in the model allows to adequately reveal the hierarchy relations existing in reality. Thus "classification" type of hierarchy relations let include alternative variants of some objects and structures' representation into the model. For example, those can reflect different experts' points of view;

t_{ho}: $O \to T_h$ are functions forming hierarchy relations among objects.

The conceptual model of executing modelling environment (software and hardware implementation) is based on the same principle (Fridman et al. 1998). Depending on the goal setting peculiarities, the implementation model may be either combined with the model of subject domain or presented by separate treelike structures. The composition of executing environment model with the subject domain model is appropriate for one-computer modelling implementation or "client-server" systems' use. Distributed modelling environment implementations require a separate model of the executing environment.

After constructing the object tree (3), (4), the resource specification begins. The resources providing object interaction must be specified first. To do so, one must define quality criteria, lists of input and output resources for every object. Forming of every input resource for every object makes a necessary condition of the model performance, so every element of lists of input resources for primitive objects must either be an external resource, or directly come from GIS, or belong to the lists of output resources of a primitive object (including the object under consideration itself). Adjusting parameters of quality criteria coming from super-objects make an exception.

External resources may come either from the SMS database (then they are assigned with the DB executor type), or directly from GIS (GISE executor type and the GIS category), or from an explicitly defined time function, or from a software module with a single input, namely time. Particularly, it is a good way to form different scenarios of input resources' variations for the modelling object. In the last case the datum is assigned with the GEN (from "generator") executor type and a name from the set E_r (17). If some scenario data are formed manually, they may also be assigned with the DB executor type. The default type for executors of output resources is TMP (temporary data), then the resources exist for one simulation cycle only. If a resource is needed for collating of several calculation variants, it must be assigned with the CMP (from "compare") executor type. By the end of link specification every input resource (excluding external data) of every model object $o_i \in O$ must be assigned with objects generating this resource and every earlier defined resource must be assigned with an executor.

Then one must specify processes generating output resources of objects in order to create the interaction relation OP (8) and the respective function op (10).

SDCM constructing ends with description of internal processes when the SDCM connectedness shall be achieved for every resource by the following actions:

the processes having output resources completely or partly declared as output object resources are assigned to those objects thus forming assignment relation OA and respective functions oa (10);

every already assigned process is assigned with objects generating its input resources thus continuing to form the relation OP (8) and the function op (10); if every input resource of a process is generated within an object the process is assigned to, the process gets the INNER category, etc, till all input resources of the last internal process in this chain will coincide with previously declared input resources of the given object;

during the process declaration, both every process and its every resource are to be assigned with executors. Besides the above-mentioned user types of executors, any resource or process may get the executor of the ES type (from "expert system"), then this resource (or every output resource of this process) will be calculated by the embedded expert system of the SMS.

In the author's opinion, the opportunity of the detailed formal analysis of the SDCM makes one of essential advantages of hierarchical conceptual models comparing with the most SDCM prototypes, particularly models of system dynamics (Forrester 1958) and situative control (Pospelov 1986).

The most important accomplishment of the conceptual modelling technology and its software means is that it allows to use the terms and notions of the subject domain familiar and clear to an expert while forming formal description of the problems to be investigated; to integrate general expert knowledge on the subject domain into a single formal model; to use automatic means for unity and completeness control of the integrated conceptual model; to automate forming of the computing experiment' executive environment.

The opportunity to apply the subject domain terminology for formal description is provided by the special conceptual model editor. While forming the model the user is just to fill in the attribute description of the conceptual model elements. The "translation" of the user description into conceptual model units is realised automatically.

Integration of the considered problem descriptions, that are presented by different experts is realised by "joining" model fragments (sub-trees) formed by them into a general model using appropriate hierarchy relations. In the course of conceptual model forming the users are provided with reference directories of the subject domain objects already included into the model as well as with electronic maps to identify the geographical situation of the described object (figure 1). These means used in model forming simplify the consequent solving of the problem to co-ordinate the presented descriptions and ensure the model unity.

Conceptual model is characterized as a declarative one imposing no sufficient limits on the range of methods and implementation means for computing experiments. Thus different modelling methods can be involved into conceptual model data processing, namely analytical, statistical, logical, simulating ones. The modelling system can easily integrate already existing reliable solutions. Above this, the conceptual modelling technology allows to identify the model components necessary to use for a certain problem solution and to specify a structure appropriate for modelling realization environment.

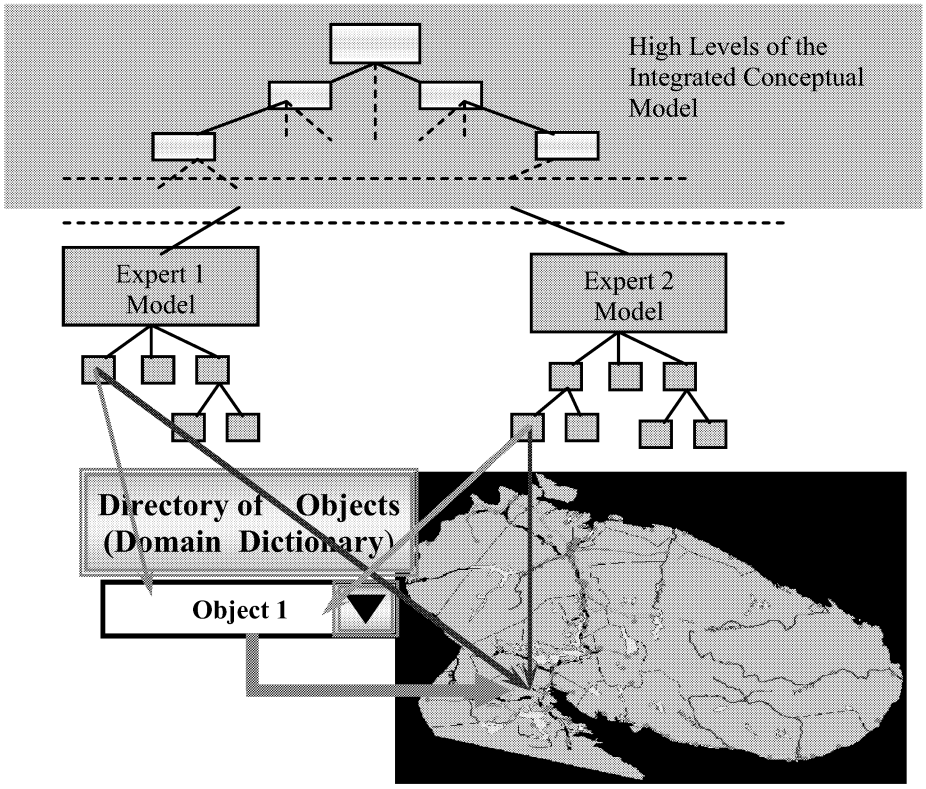


Figure 1. Integration of different experts' models into a common conceptual model of the subject domain.

Choosing of the conceptual description fragment relevant to a certain problem is delivered by a model graphical editor as well as by certain algorithms intended for complete definition (extra-definition) of the single solution structure. The editor marks certain nodes (initial, goal, intermediate ones) on the model graphs. Then some extra-definition procedures based on the algorithms analogous to the solvability analysis algorithms follow to extra-define the set of model elements that will provide the declarative solvability of the given problem taking the set up initial conditions into consideration.

The marked conceptual model fragment is the basis to form specification of the executing environment for the problem presented, that is conditioned by the executors assigned for the marked conceptual model elements.

The conceptual model structure as well as its software means enable one to form a common model of an NTC of ready subsystems' models. Similar in formal structure processes related to different problems or objects of the subject domain, require one executor according with the reuse principle for standard solutions.

The access to the integrated SDCM enables a user to find the earlier worked out fragments meeting his purposes.

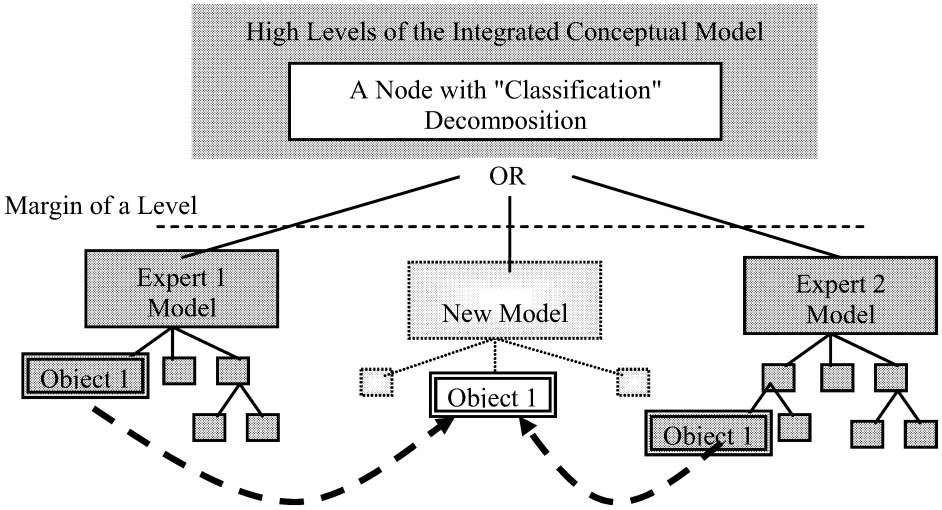


Figure 2. Forming of a new model by partial or total "lending" of earlier defined features of an object.

Moreover, a decision-maker (DM) is apt to develop his purpose models using models worked out by experts in various subject domains. Thus knowledge about different features of any object declared in the conceptual model of the subject domain and considered by experts within the frame of their certain problems is integrated. If the same model object appears in the object tree twice as a result of its super-object decomposition on different classifiers (OR-decomposition type object on figure 2), the synonym technique is used to preserve formal accuracy of the SMS model (Fridman 2000), i.e. a new quasi-object is generated with an identifier one-to one depending on the initiate object identifier. The further DM' modelling may involve into the new model any resources connected with both synonyms. Then generation of executing environment for the new model research is accomplished automatically.

CONCLUSIONS

The developed situative modelling system is intended for automation of every modelling stage, wide usage of expert knowledge, employment of the embedded GIS not for object mapping only, but for task setting, spatial-dependent calculations and modelling results' displaying as well. The SMS provides a DM with a support to ground altering (or not altering) the structure of his subordinated object and substitute an expert council in this sense.

The situative conceptual model of an NTC is the core of the modelling system. It is intended to formalize and integrate the general expert knowledge of the subject domain and dynamics of its development.

The worked out modelling system provides using different methods for investigation not-enough-formalized complex non-stationary spatial objects, ensures complex use of expert knowledge to form criteria and chose alternatives for more detail studying in the simulation modelling mode.

Technologically the SMS is to narrow the gap in methods of organizational systems' modelling within the structural approach to the open subject domain model implementation. We have proved that the SMS allows to model every main hierarchy type considered by the general theory of hierarchical systems (stratified, multilevel and organizational hierarchies (Mesarovic et al. 1970)).

REFERENCES

Forrester, J. 1958. Industrial Dynamics - A Major Breakthrough for Decision-Makers. In: *Harvard Business Review*, Vol. 36, No. 4, pp.37-66.

Fridman, A. 1998. Situative Approach to State Modelling of Spatial Object. In: *Informational Support Systems for Regional Development*, Apatity: Kola Science Centre, pp. 45-49 (in Russian).

Fridman, A. 1998. Situative Approach to State Modelling of Spatial Object. In: *Informational Support Systems for Regional Development*, Apatity: Kola Science Centre, pp. 45-49 (in Russian).

Fridman, A. 1999. Classyfiing of Situations and Alternative Model Structures' Collating in Situative Computing Experiment System. In: *Simulation in Regional Development Research*, Apatity: Kola Science Centre, pp.14-24 (in Russian).

Fridman, A. 2000. Generation and Sequence Relations Using for Automatic Analysis of Conceptual Model. In: *Theoretical and Applied Models of Regional Informatisation*, Apatity: Kola Science Centre, pp.75-83 (in Russian).

Fridman, A.; A. Oleynik and V. Putilov. 1998. GIS-based Simulation System for State Diagnostics of Non-stationary Spatial Objects. In: *Proceedings of 12th European Simulation Multiconference* (ESM'98), Manchester, UK, June 16-18, 1998, vol.1, pp.146-150.

Mesarovic, M. and Y.Takahara. 1975. *General Systems Theory: Mathematical Foundatios*. Acad. Press, New-York, San Francisco and London.

Mesarovic, M.; D. Macko and Y.Takahara. 1970. *Theory of Hierarchical Multilevel Systems*. Acad. Press, New-York and London.

Pospelov, D. 1986. *Situative Control: Theory and Applications*. Moscow: Science (in Russian).

Yourdon, E. 1975. *Techniques of Program Structure and Design*. Englewoods Cliffs, NJ: Prentice-Hall.

Yourdon, E. 1989. *Modern Structured Analysis*. Englewoods Cliffs, NJ: Prentice-Hall.

FLEXIBLE HIERARCHICAL ARCHITECTURE OF SIMULATION MODELS

Norbert Adamko
Valent Klima
Michal Lekýr
Faculty of Management science and Informatics
University of Žilina
Univerzitná 8215/1, 01026 Žilina
Slovak Republic
E-mail: Norbert.Adamko@fri.utc.sk
Valent.Klima@fri.utc.sk
lekyr@utcpd.sk

Antonín Kavička
Jan Perner Transport Faculty
University of Pardubice
Studentská 95, 53210 Pardubice
Czech Republic
E-mail: Antonin.Kavicka@upce.cz

KEYWORDS
Agents, Discrete simulation, Simulation architecture, Model desing, Hierarchical, Software engineering

ABSTRACT

This paper deals with the ABAsim architecture of simulation models, which provides flexible base for creation of simulation models of large service systems. Architecture is based on cooperating agents organized into hierarchical structure. ABAsim architecture provides great flexibility to the designer allowing to exchange agents, and/or their components, together with the support for distributed execution of simulation models. Specialized CASE tool for design and creation of simulation models developed under the architecture is also discussed.

INTRODUCTION

Architecture *ABAsim* was developed mainly for simulations of large service systems. A *service system* is understood as a system focused on elaboration of *orders* (attendance of *customers*) and execution of *services* related to them. The mentioned services can further initiate another set of orders. The service systems include a wide class of various systems e.g. factories, any transportation and logistic nodes/junctions, hospitals, repair shops etc. The natural and technical systems do not belong to that domain.
Let us mention the most important features of service systems, which essentially influenced the architecture properties:

- The service system structure can be considered (from the viewpoint of order elaboration) as strictly *hierarchical*. The order (the customer) entering the system (e.g. produce a car, treat a patient) calls recursive sequence of suborders according to the rules of competence redistribution.
- All system elements (subsystems) work synergically, unlike the majority of natural systems, with the common goal to elaborate the order. Thus, the architecture is not determined to work with the processes of the following kinds: evolution, competition and parasitism.
- The entities of a service system (orders/customers and resources) can be divided into specialized classes with the same behavioral rules for included entities. It means that the responsibility for the behavior of entities is taken by their superior subjects (agents) and hence there is no reason to consider entities as agents.
- The service systems represent usually large-scaled systems. It is mostly necessary to transfer service resources to the customer (or vice versa), in order to realize the service activity. Hence, there are typical frequent complex transposition processes within those systems.

COMMUNITY OF AGENTS AND ITS STRUCTURE

The simulation models of simple real systems could be composed of only one agent, however the simulations of complex service systems are obviously connected with *multi-agent approach* using the agents within some organizational structures. Let us remark that the philosophy of ABAsim architecture was also partly inspired by the paradigm of reactive agents, which is based on the society of rather reactive than proactive agents. The intelligence of that society *emerges* when observing the whole community and not the separate (relatively little intelligent) members.

Multi-agent hierarchical system can be demonstrated on the following example (Fig.1), where A_0 (agent of transportation network) divides the management of the whole network S_0 to two partial managements of regions S_1 and S_2. We say that agent A_0 *delimits* the network management to two peer regional agents A_1 and A_2, which inform their *boss* about the important facts connected with the whole network. Agent A_0 can also send important supra-regional information to its *subordinates.*

It is expected that both mentioned regions carry out the same kinds of management operations. Each of regional agents further utilizes the set of subordinated specialized agents in order to fulfill defined tasks. It means in fact that

regional agents *delegate* some portion of their competences to hierarchically lower ranked colleagues. It is obvious from the mentioned figure that agent A_1 utilizes different structure of subordinates than its colleague A_2. Another point of view is that agent A_0 is a representative (boss) of the whole model M_0 (simulation model of transportation network). That fact is denoted as: $A_i = boss(M_i)$ and $M_i = model(A_i)$. The models M_1 and M_2 represent sub-models of model M_0. The models on the lowest hierarchical level (the leaves) are always realized as one-agent models. Let us finally denote the structure of the model M_0 in the form of an algebraic expression $[M_1(M_3,M_4,M_5),M_2(M_6,M_7(M_8,M_9))]$, where the pairs of brackets encapsulate the models, which were created as an act of delimitation and the pairs of parentheses encapsulate the models with delegated competences.

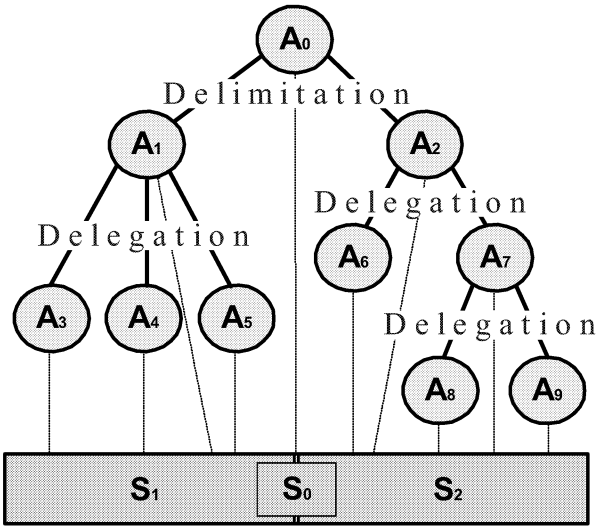


Figure 1: Hierarchical structure of agents

MODULARITY AND MODEL RE-USE

Structure of a simulation model as well as of individual agents represents such features, which highly support a flexibility of simulation model within ABAsim architecture. In spite of that fact the message-oriented communication mechanism does not enable such level of flexibility, which would allow changing a sub-model structure without a need of additional changes within other model parts.

For example, if the structure of sub-model $M_1(M_3,M_4,M_5)$, illustrated on Fig.1, is supposed to be changed to $M_1(M_3(M_{10},M_{11}), M_4,M_5(M_{12},M_{13}))$ then within other model parts: $M_0[*,M_2(M_6,M_7(M_8,M_9))]$ could be needed to modify identifiers of addressees (agents) hierarchically subordinated to M_1. The reasons, why to change a sub-model structure, can be motivated e.g. by an additional decision to implement that sub-model on the more detailed level.

Inter-agent communication

Let us present basic principles of solving the above-mentioned problem within ABAsim architecture profiting by hierarchical organization of sub-models. The model $M = \{M_0,M_1,M_2, ..., M_S\}$ represents a set of sub-models reflecting a set of agents $A = \{A_0,A_1,A_2, ... , A_S\}$. For all $I = 1...S$ we denote:

- $I_{Ag}(A_i)$ – a set of messages directly addressed to an agent A_i, which is able to respond to them (a set of *direct-messages* of an agent A_i),
- $I_{Sub}(A_i)$ – a set of direct messages of all agents, which are subordinated to A_i (a set of *mediated-messages* by an agent A_i),
- $I_{Mod}(A_i) = I_{Ag}(A_i) \cup I_{Sub}(A_i)$ – a set of messages, which can be elaborated by a $model(A_i)$ regardless of its structure (a set of *model-messages* in respect of $model(A_i)$).

Each message kind can be utilized in three following variants:

- Message of kind $mess(A_i, A_j, agent)$ is sent by an agent A_i to an agent A_j, which accepts it only if $mess \in I_{Ag}(A_j)$ – it is a standard *addressed message*.
- Another message type $mess(A_i, A_j, model)$ is dispatched by a sender in case it is not known a concrete target addressee/agent (or due to flexibility it is not desired to be determined). However, it is evident that $model(A_j)$ is qualified for its elaboration. That message is treatable if $mess \in I_{Mod}(A_j)$ and is called *partially addressed message*.
- Finally a message kind $mess(A_i,*,*)$ represents a *non-addressed message*, which is elaborated in case that $mess \in I_{Mod}(A_0)$.

If a non-addressed or partially addressed message is supposed to be delivered then a special *addressee searching algorithm*, based on systematical investigation of sets $I_{Mod}(A_i)$, is automatically started. Special treatment is applied to a case of an addressee from a set of delimitated agents, which receive the same kinds of model-messages (additional selection criterion related to such messages can be used).

Thus, it depends only on a model designer if he/she prohibits for a selected sub-model to directly address its agents (except *boss*) – such a command enables making safe structural changes/modification of that sub-model. Total flexibility is reached only if non-addressed messages are allowed, however that concept is connected with more time-consuming demands.

Agent decomposition

Each of agents can be decomposed to four following groups of *internal components*:

a) The first *control* and *decision making* component, called *manager*, is responsible for making decisions

and inter-agent communication. In addition, the manager represents the central agent component because it initiates the work of other internal components and also can communicate with all of them.

b) The group of *sensors* is specialized to information mining from the state space. That group is composed of two kinds of components - the *query* delivers the required kinds of information instantly and the *monitor* makes scanning of state space during some time interval and continuously brings important pieces of information to the manager.

c) The next group of *solvers* provides problem solutions for the manager, which can accept them or asked for alternative ones. The *advisor* is passive component that is able by return to react only to the manager's request for delivery of problem solution proposal. The typical forms of an advisor can be represented e.g. by optimization algorithms, neural networks or human operator. On the other hand the *scheduler* (focused on the restricted scope of problems) works for manager continuously on the base of either a priori created rosters/schedules (e.g. connected with allocations of resources) or making its own dynamic forecast for defined time interval.

d) The last component group includes *effectors* (actuators), which make changes of system status after getting corresponding instructions from the manager. The other kinds of agent components are not allowed to make the mentioned changes. The *action*-component makes instant state changes (e.g. switch a traffic light over, close the train doors), while the *process* (e.g. the crane movement) makes them continuously till its tasks are finished.

The effectors, sensors and solvers are briefly called manager's *assistants* and can be further distinguished as:

- *Continual assistants*, activities of which take some interval of simulation time (processes, monitors and schedulers).
- *Instant assistants* (actions, queries and advisors), which are active only within one instant of simulation time.

Model Builder

The process of creating complex simulation models based on the ABAsim architecture is a highly challenging task. When designing such models the design complexity increases accordingly. Using traditional methods of model design introduces high risk of making errors; therefore the ABAbuilder tool which was created upon the existing architecture brings a great improvement in this process. ABAbuilder is a CASE tool based on the ABAsim architecture and it supports several aspects of model development including system analysis, design of communication, use of methodologies, prototyping, model maintenance and re-engineering.

One of the greatest contributions of the architecture is the flexibility of model configuration. ABAbuilder is capable of creating large palette of alternative internal components, which can be used to construct the simulation scenario. The experimenter is allowed to change executive characteristics of each of the system agents by using different components with pre-programmed behavior and decision making algorithms. Even less experienced programmers can design their own experiments. This is due to similarity of modeled system with the real world.

A simulation model can be described by the hierarchy of system agents. From the ABAbuilder point of view, not only agents, but also their internal components are considered as model building base. There are several phases each model has to undergo, before experiments can be carried out:

1. design of the hierarchy and communication of system agents
2. implementation of the simulation model
3. verification of the model
4. parameters setup
5. source code generation

To be able to accomplish these particular phases the CASE tool had to be divided into four interacting modules: The ***visual environment*** displays the simulation model from different perspectives. Its main purpose is to display the agent hierarchy and internal components of each agent. It allows users add new agents to the hierarchy, edit their internal components and define the message flow inside the model. Visual Petri nets editor is used to define the reaction of agent's manger to received messages. Users are allowed to look at all aspects of the hierarchy at once, or they can apply various sets of filters.

The ***error localization*** module works continuously and it checks the model each time a change has been applied. It provides user with two kinds of messages: *error messages* and *warnings.* Errors messages appear when user tries to do something which is not allowed by the methodology of creating models based on the ABAsim architecture. Warnings inform users of possible unwanted problem solutions.

When the decision is made by the user that the model is ready for deployment the ***source code generation*** module comes into use. This module is capable of creating the framework of the simulation model. It generates source files in chosen destination language (only Object Pascal is currently supported). These files contain all agent classes, their internal components, communication and the hierarchy. User then finalizes the simulation model in external development environment (e.g. Borland Delphi). Libraries containing the simulation kernel, which allow compiling a standalone application, are provided.

To be able to reconstruct the model from source files to its graphical representation a ***source code parser*** module is necessary. This module analyzes given files, which where previously generated and it provides data for the visual environment.

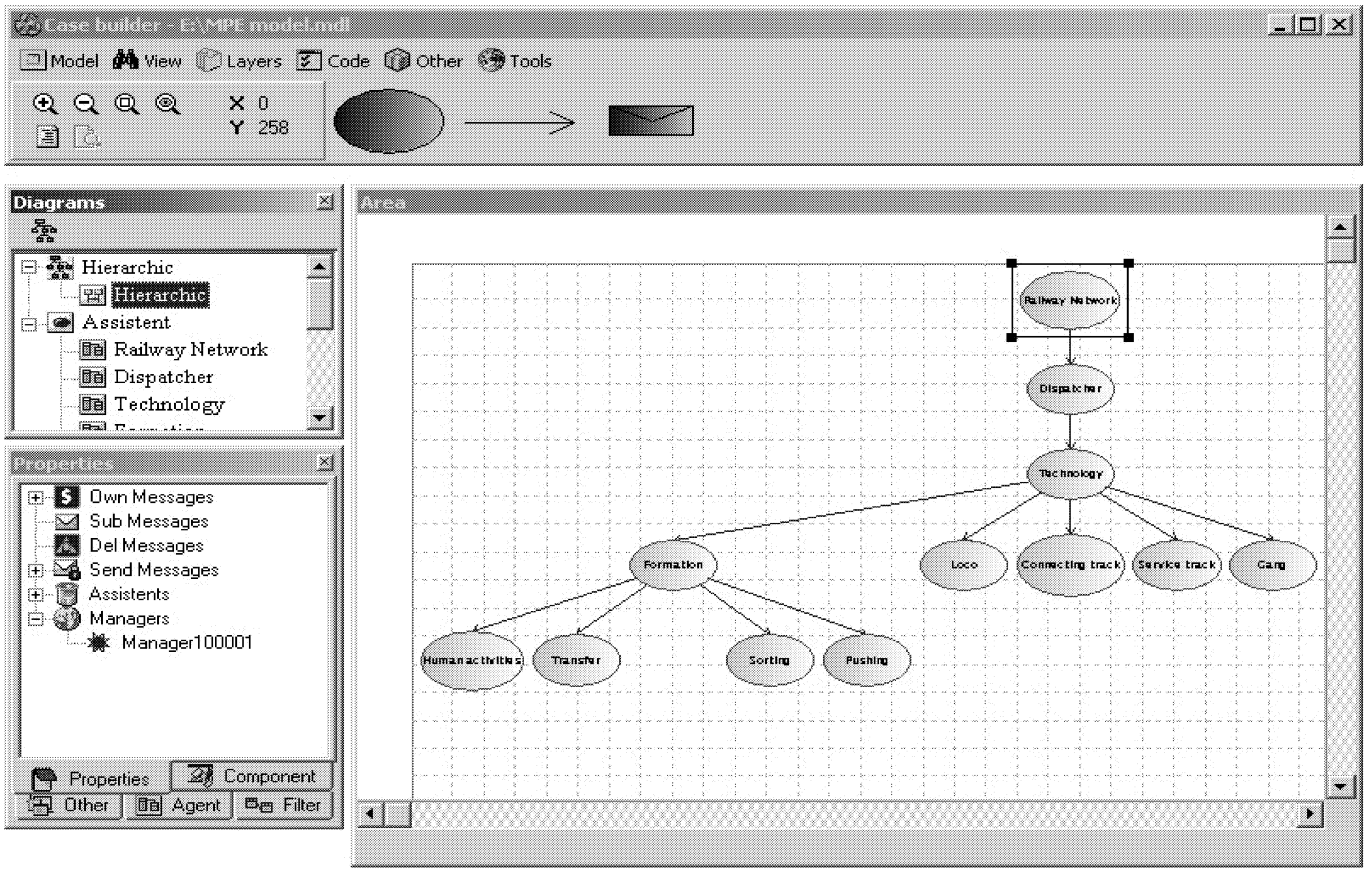


Figure 2: Screen capture of the ABAbuilder tool

ABAbuilder tool enables to construct new models and modify existing ones with the help of prefabricated model parts, which are selected from bases of agent components (assistants, managers), agents, sub-models and their various configurations. It makes the whole process of modeling much easier to accomplish and more understandable for more as well as less experienced users.

DISTRIBUTED SIMULATION UTILIZING HIERARCHICAL STRUCTURE OF AGENTS

Complex simulation models of large service systems are often very computationally demanding. Distributed simulation models, which utilize power of more computers, could be one of the solutions to this problem. Distributed models are valuable also in situations where more users-experimenters would like to interact with the simulation model during its execution.

ABAsim architecture provides the possibility of flexible distributed execution of simulation models. Sub-models (consisting of one or more hierarchically organized agents) can be assigned to different processors for execution. The hierarchical structure of agents is exploited for the time synchronization of the sub-models.

Every agent only synchronizes its execution with parent and directly subordinate agents. Used synchronization algorithm is based on conservative approach to time synchronization, which was preferred over optimistic approach due to various reasons (e.g. implementation complexity). Important feature, which provides great flexibility, is that all agents have exactly the same time synchronization abilities.

Each agent is responsible for managing its local time and controls execution of subordinate agents, which can start execution only after receiving permission from parent agent. Agent provides to its parent the actual time stamp of future event in its sub-model. Agent on the top of the hierarchy is the only one eligible for changing the global simulation time. After the global time change, the boss agent of the model sends run permission message to its directly subordinate agents, sub-models of which can safely execute (based on their future event times and lookahead value). Every agent after receiving run permission message checks its directly subordinate agents (using the same algorithm as any other agent, including boss) and forwards the message to agents, sub-models of which can safely execute - this way the run permission propagates to all agents, which are able to execute at current time.

USE OF THE ARCHITECTURE

As a bright example of architecture possibilities and flexibility, simulation tool Villon (Klima et al. 2001, Koenig 2001) can be mentioned. Villon is detailed

microscopic simulation model of logistic junction operation with predominant transportation processes. The junction is understood to be e.g. railway marshalling yard, railway station for passenger transport, factory sidings, airports, container terminals, etc. Utilizing the ABAsim architecture, Petri nets and network graphs Villon tool is flexible enough to model different means of transportation considering their specific attributes. Using Villon one can build simulation models of railway marshaling yards with only railway traffic, but also combined models, with inter-modal operation. Villon was successfully used in commercial environment to provide services for many clients, among them are national railway companies from Germany, Austria and Switzerland, private companies BASF, Volkswagen and many others.

The ABAsim architecture is also being used in master degree course "Discrete simulation" at the University of Zilina. Long term experience proved that the architecture is flexible and robust base for development of simulation models of large service systems.

REFERENCES

Jennings, N. R. 2001. "An agent-based approach for building complex software systems", *Communications of the ACM*, Vol. 44, No.4, pp.35-41

Kavička, A.; Adamko, N.; Klima, V. 2000. "Simulation support for railway junctions infrastructure and processes economization", In: *Proceedings of MI 2000 conference*, Zilina, Slovak republic, 2000

Klima, V.; Kavička, A. 2000. "Simulation support for railway infrastructure design and planning processes", In: *Proceedings of COMPRAIL 2000 conference*, Bologna, Italy, 2000, pp.447-456

Klima, V., Kavička, A., Adamko, N. 2001. "Software tool VirtuOS – simulation of railway junction operation", In: *Proceedings of ESM 2001 conference*, Prague, Czech republic

Koenig, H. 2001. "VirtuOS – Simulieren von Bahnbetrieb", In: *ETR–Eisenbahntechnische Rundschau*, Hestra-Verlag, Januar/ Februar 2001, pp.44-47

Wooldridge, M.; Jennings, N. 1995. "Intelligent Agents: Theory and Practice", *The Knowledge Engineering Review 10 (2)*, pp. 115-152

AN OPEN SOURCE OBJECT-ORIENTED ARCHITECTURE FOR DISCRETE EVENT SIMULATION

L. Castelluzzo, A. Grieco, F. Nucci, S. Zacchino, A. Anglani
Università degli Studi di Lecce
Dipartimento di Ingegneria dell'Innovazione
via per Arnesano, 73100 Lecce, Italy
E-mail: francesco.nucci@unile.it

KEYWORDS
Computer integrated manufacturing (CIM), Model design, Software engineering, Qualitative models, Object-oriented models.

ABSTRACT

Simulation is a valuable tool enabling organizations to investigate possible strategies for improving performances and reducing operating costs.
In this work, an Object-Oriented Approach (OOA) to create simulation models and the relative open source software architecture are introduced. The goal of such an approach is to reduce model composition time by means of a direct system-model mapping. The resulting software has the following features: code reusability, modularity and extensibility. In particular, extensibility has a key-role to satisfy some common needs in the field of simulation model development. Moreover such an approach permits to perform fuzzy simulation run, consisting in modelling and simulating systems whose parameters are defined in uncertain form and no statistical distribution can be inferred because of the lack of data.
In such a paper a description of the open source project is made, a comparison is presented between the presented methodology and other open source simulator projects in order to explain the main differences. Finally, an example in the manufacturing field is presented to show the potentiality of our approach.

INTRODUCTION

System engineering is an important activity in various fields. Several tools are required to support solution designing and alternative assessing. Computer simulation presents a fast and inexpensive way to reach such a goal. Moreover, the increasing complexity of modern systems makes computer simulation essential to perform a complete analysis. Higher integrated and automated production systems need a detailed and careful analysis of the dynamic scenario in order to prevent misinvestments. Moreover, the planning tools should be appropriate and efficient to achieve best results. Computer simulation in manufacturing is often used to find the optimum configuration of a multi-station process or an entire factory in order to achieve specific goals. For example, such goals are the minimum capacity of buffers or the minimum number of machines, personnel and fixtures for a given production output. When using the methodology of computer simulation, these optimization procedures are iterative. Simulation analyst develops the model, runs the simulation, analyzes the output data and modifies model input data relatively to the goal, all based on his knowledge and experience. He/she runs the simulation cycles until the results are sufficient. Different techniques can be adopted in order to speed-up the searching. Such algorithms are described in the field of simulation/optimization theory. The important view is, that the simulation analyst can choose between different optimization techniques to achieve best results in a given application. For this reason, it is important to integrate computer simulation with optimization techniques. Open source paradigm allows researcher to select the suitable paradigm of computer simulation and optimization techniques. Using commercial software packages, such integration is more difficult because of limited software interfaces available.
Planning complex production systems and optimizing existing processes need suitable simulation tools (Hlupic and Paul 1999). In recent periods, designers faced to the problem to supply accurate information promptly on analyzed systems. Therefore, analysts need to create simulation models in short time. In order to design a suitable modeling process, it is necessary to carefully select entities and parameters to be used in the simulation. A definite necessity for utilities assembling the simulation model without programming effort is also present (Oloufa et al. 1998). Such tools permit users to define the system in familiar terms rather than in terms of simulation languages. Moreover, it is possible to check user's responses for consistency and completeness. The designing phase of a simulation model often requires the following utilities: creation of new models adapting the existing ones, visual representation of the system decision-makers friendly and high-level simulation constructs at user level to simplify the coding. Furthermore, simulation execution has to provide data for both performance assessment of analyzed systems and validation of designed models.
Many efforts have been made in order to apply the successful concepts of object-oriented analysis and design (Booch 1993) to simulation theory, especially in commercial products such as ModSim (CACI 1993) and Simple++. Free Open Source tools are also available, but only in very specific fields of application. None of these products, anyway, can be regarded as a complete and general-purpose framework allowing analysts to model real world cases by applying the OOA concepts. Often, the modeling approach required for computer simulation software is not conformed to the real system to be reproduced and analyzed. For this reason, an abstractive effort is required to describe systems in a different way they appear. Instead, the OOA has proved uncommon representative capabilities and has brought to the simulation field the advantages already provided to software engineering (Boehm 1981), (Gamma et al. 1994).
In this paper we introduce the software architecture, adopted by the Discrete Event Object-oriented Simulator (DEOS) project, that satisfies the requirements cited above. We consider other important open source simulator projects that already faced the question, and present innovative aspects introduced by DEOS. In particular, we present DEOS tools

making possible the composition of simulation layouts (a set of connected objects representing a real system) and the data collection. Finally, an example of DEOS modelling in the manufacturing field is described and main results are showed.

DEOS PROJECT

A review of open source simulation projects

In the presented work, we adopted the open source simulation project directions (Open source website) in order to allow reading and modifying the DEOS source code. Since DEOS is part of a research work, the open source approach is more suitable than classical closed source model. A list of some open source projects creating object-oriented simulation architectures is reported in the following.

OMNeT++ is licensed under terms similar to those of the GNU General Public License for non-commercial use (OMNet++ website). It can run discrete event simulations and has shown to be suitable for simulation of communication networks and hardware architectures. OMNet++ architecture (Varga 2001) includes components written in C++ that can be merged in models using a higher level language referred to as NED. A compiler translates a NED file in C++ and the resulting file is compiled and linked with the simulation program. A graphical editor of NED files is currently under development. In the future, XML language will be used beyond the current NED files. OMNet++ is available on different platforms and is modular since its kernel can be embedded into other applications as simulation engine.

Network Simulator (NS) is a discrete event simulator, specific for network simulations (Breslau et al. 2000). It is mainly used in research contexts and its extensibility allows for a ceaseless growth of modeling capabilities. Though it is written in C++, the Tcl language can be used to create simulation layouts by using existing classes of the NS simulation library. Validation tests are provided with NS for many network protocols. For NS documentation see (NS website)

SIMOO is a strongly object-oriented environment for discrete event simulation (Copstein et al. 1997). It has been successfully used in various contexts: real-time systems, embedded electronic systems and processor architectures. It makes a separation between the object-orientation, used to design and to implement simulations, and the modeling paradigms, used to describe the models. Indeed, the model can use different communication protocols and behavior description styles. Simulation layouts are built by using a visual editor that shows two kinds of specific diagrams: class and instance diagrams. The class behavior can be set by writing C++ code directly into the editor. When the composition phase is complete, the editor submits the classes to a C++ compiler which generates an executable file. This file runs the simulation and shows the results. Project documentation is available in (SIMOO website).

C++SIM consists in a package of object-oriented libraries for discrete event process-based simulations written in C++ (Arjuna-Team 1994). It provides facilities like queueing algorithms and random number generators as in SIMULA language. Furthermore, it includes: entities manipulation facilities, statistical routines and debugging classes, see (C++SIM website).

OOSIM stands for Object-Oriented Simulation in Manufacturing, (Govindaraj et al. 1993). The free source code part is a library of general simulation facilities. In (Bodner et al. 1994) an application to a manufacturing system is reported, but there is no reference to the resulting source code. For OOSIM documentation see (OOSIM website).

The DEOS modeling approach

The DEOS simulation software takes advantage of the object-oriented approach. Current version of DEOS allows complex system modeling in compliance with typical simulation theory concepts, expressed in terms of *resources*, *entities* and *events*. An *environment* stands for a set of simulation layout components. If necessary, resources can exchange entities in accordance with a communication protocol. Each resource has proper *attributes* (e.g. initial parameters). During simulation runs, the resources can add attributes to the entities, and such attributes can be read or modified by other resources. The resources can be also linked with *pseudo-random variables* having assigned probability distributions.

Simulation evolution is obtained by means of an event list called *timeline* allowing resources to schedule events in proper instants. Resources interact with each other by using a communication protocol: each resource has input and output ports and each port is identified by a number. Users can connect output ports to input ports of different resources. Resources can use such port connections to exchange entities and read attributes. DEOS project documentation is described in (DEOS website).

The architecture

Figure 1 and Table 1 show a part of the DEOS class architecture. The class TDLResource is the one from which new resources should derive their logic class. Figure 2 and Table 2 show an environment that includes the above plugins.

Description

Modularity received a particular attention in DEOS. Indeed, the architecture involves use of external modules - *plugins*. These plugins are dynamically linked to the main executable to add features such as new resources, data acquirers, environments, etc. Plugins definition files describe plugins by means of proxy classes. Figure 3 and Table 4 show the part of DEOS architecture used for the plugin management.

Besides the well-known advantages, related to maintenance and reuse of software, such architecture shows a notable degree of extensibility. Moreover, this solution has not a negative impact on execution time. Plugins may contain the implementation of new resources, useful for specific system modeling, or other simulation tools.

DEOS is currently available for Microsoft Windows platform. However, the software has been developed to be easily ported on different platforms. For this reason, the code has been divided in two parts: logic-portable code and platform-related code (capable of handling graphical user interface).

Users that need to create a simulation layout with DEOS, can use a user-friendly interface allowing to select the necessary plugins. Indeed, in the highest layer of the architecture a visual editor of simulation layouts is available. First, users select the suitable environment; after they put resources and other simulation tools on such environment. The user interface has typical commands to improve personal productivity, such as resource aggregation and "cut & paste" operations.

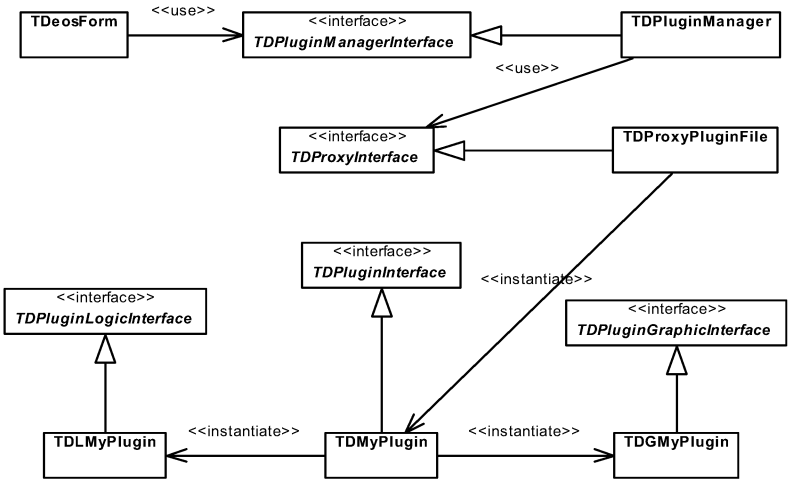


Figure 1: Main Classes of DEOS Architecture

Table 1: Description of the main classes of DEOS architecture

Class	*Description*
TDeosForm	The main application.
TDPluginManagerInterface	Abstract class acting as interface for the plugin manager.
TDPluginManager	It loads all plugin files and extracts plugin data. It can create a plugin instance.
TDProxyInterface	Interface for the proxy of a plugin file.
TDProxyPluginFile	A specific proxy for the "PluginFile" plugin file. It provides information about plugins and creates new plugin instances when the plugin manager needs them.
TDMyPlugin	The main class of "MyPlugin" plugin. It holds plugin information such as: type, name, category. It owns pointers to logical e graphical classes.
TDPluginLogicInterface	The interface of each logic class.
TDLMyPlugin	Logic class of "MyPlugin". It handles the logic aspects of the plugin.
TDPluginGraphicInterface	The interface of each graphic class.

Another feature of DEOS architecture is the interoperability with other applications. The use of a XML standard to describe simulation environments and associated plugins, allows other applications to access to parameters and results of a simulation run. For example, simulation optimization tools can use such a feature to create specific layouts to be simulated and to get results. Moreover, developers can write interfacing plugins able to communicate with other external applications and DEOS components. Such external applications can be legacy simulators, statistical tools and simulation optimization tools.

Data collection and simulation results

Data the users need to collect are unknown *a priori* to developers of new resources. Users can only provide some information having high possibility to be desired. For example, user needs the number of parts processed by a machining center, the flowtime related to entities, etc. For this reason, developers have to re-write the user interface to access the data for each additional resource type. Whereas, a special plugin type for the collection of any data coming from simulations object is available in DEOS. These plugins are referred to as *data acquirers* and are classified in *stat collectors*, *controllers* and *gauges*.

The above classification is based on the purpose the collected data are used for. In particular:

- *stat collectors* use collected data to achieve statistical analysis usually shown when simulation ends;
- *controllers* can change simulation parameters when a conditions on the collected data become true;
- *gauges* show a multimedia feedback for the current simulation run.

A general method has been defined in order to specify the information that data acquirers have to collect. A common user interface to configure data acquirers has been created for this purpose:

- data acquirers are provided with *inputs* which can be used to establish data to be acquired;
- each input is assigned with a *warner* which is the simulation element that generates a data acquisition from the input.

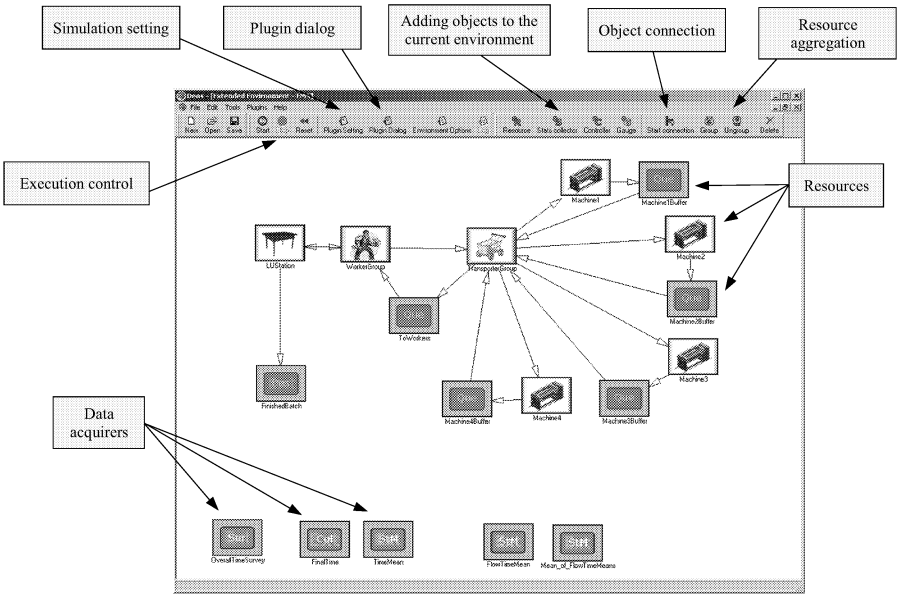


Figure 2: An Example of DEOS Environment

Table 2: Description of DEOS Environment

Class	*Description*
Execution control	These buttons start, pause, stop and reset the execution of the current simulation run.
Simulation Settings	It opens the object inspector that allows users to set object properties.
Plugin Dialog	Each plugin can have a dialog box to show. This button opens it.
Adding object to the current environment	These buttons open a dialog box with a list of plugins that can be added to the current environment.
Object connections	It allows creating connections between resources.
Resource aggregation	These buttons group/ungroup resources.
Resources	Resource instances
Data acquirers	Data acquirer instances linked to simulation data sources.

Scripting languages and DEOS

DEOS is based on the C++ language and such standard has to be followed for the plugins DEOS coding. This language has permitted to create fully object-oriented software, able to reach high performance levels. However, using this language only requires a notable effort for creating new plugins to the designers that are not able to use C++. This problem has been solved by introducing a new type of plugins - *scriptable plugins*. Such plugins can be used together with the usual C++ compiled plugins. Indeed, we exploit the architecture modularity to create a resource plugin that can be set using a script in Tcl language (Tcl website). The same technique can be used to integrate any other scripting language. In other words, when a resource is added to the environment, such resource is in "blank" state: no features and no behaviors are defined. Nevertheless, it is possible to provide the resource with attributes, random variables, input ports, output ports and a behavior by means of a Tcl script. The act of assigning all this properties to the "blank" resource is done when, at runtime, the user creates a simulation layout (instead at compile-time as usual).

Other simulation packages allow to create simulation layouts using script languages like Tcl, but unlike of DEOS, it is not possible to modify the objects that participate in the simulation run. Table 3 shows an example of the difference between the "DEOS approach to Tcl" and the ones of other simulation packages.

Table 3: Use of Tcl in DEOS and in other Simulators

Tool	*TCL description*
DEOS	1 Object X executes the event Y on time t 2 On event Z, object X creates an entity 3 Object X generates a pseudo-random number and uses it for ...
ARENA, NS, ...	1 Put n object of class C on the layout 2 Connect object X to object Y 3 Let the attribute y of the object X be ...

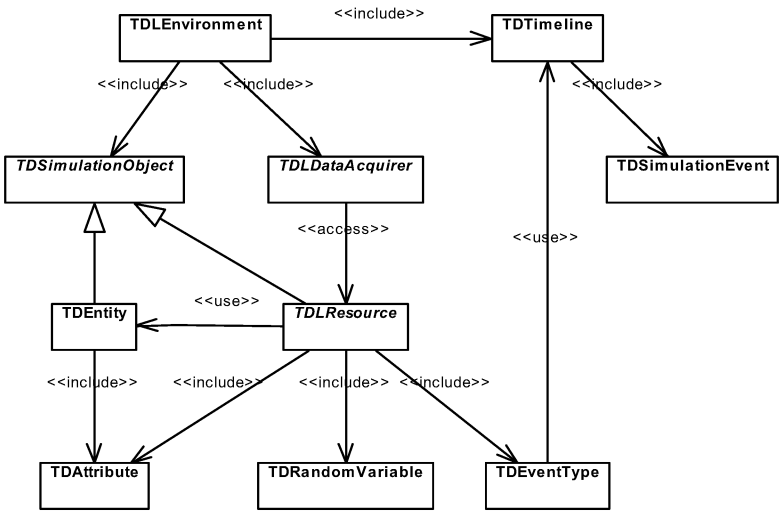


Figure 3: Classes Involved in Plugin Management

Table 4: Description of classes involved in plugin management

Class	*Description*
TDLEnvironment	It links the plugin instances participating in simulation. It provides the plugin with services such as random number generators and entity instantiation. It can be used to start, pause, stop and save the simulation run.
TDTimeline	The event list to be execute. It can be used to schedule events at a given time. It controls the simulation execution.
TDSimulationObject	This is the ancestor class for any resource or entity. It allows the handling of simulation object attributes.
TDLDataAcquirer	Data acquirers should derive their logic class from this one.
TDSimulationEvent	Each object of this class represents a single event that the timeline should execute at a given time.
TDEntity	The implementation of a DEOS entity.
TDLResource	This class should be derived to create the logic class of resources.
TDAttribute	The implementation of a resource/entity attribute.
TDRandomVariable	The implementation of a resource random variable..

The main advantage introduced by this plugin is the possibility of using an interpreted language. At the same time, this language is simpler than C++ and extensible. However, the script interpretation engine may cause loss of performance during the simulation execution; nevertheless a new resource can be defined in short time.

Fuzzy uncertainty

Simulation theory deals with process variability and leads to collect information about system dynamics. Up to now, in classical simulation approaches, variability is only considered from the stochastic point of view, even if it may be defined in different ways based on the available information and the relative degree of certainty. When it is possible to describe system variability by means of statistical distributions, classical simulation tools are able to model manufacturing systems thoroughly and they may be considered exhaustive in the solution of management problems. Otherwise, if variability can not be described by statistical distributions, as a consequence of the lack of experimental data (e.g. a new system adopting innovative technology), simulation methodology can deal with such a problem only if further questions involving *qualitative models* and *fuzzy sets* are considered (Fishwick 1991). In DEOS, such concepts have been applied to discrete event simulation to model uncertainty in the input data concerning event occurrence time. In other words, processing time and arrival time of entities can be modelled as uncertain data and are represented by using fuzzy set theory. Layouts are assumed not to be subject to such a problem.

Various approaches to fuzzy simulation (Azzaro et al. 1997) (Grieco et al. 2003) can be used in DEOS. Since event occurrence times are subject to uncertainty, the key points are how to manage the simulation event list and how to update the fuzzy simulation clock. These two tasks are mainly based on the ranking algorithm of the fuzzy sets representing the event occurrence times (Yuan 1991) (Liou and Wang 1992). DEOS architecture allows developers to modify the definition of *event list* and *occurrence time*, in order to cope with fuzzy simulation approaches. Such features are not present in any of the simulation software previously reported. Indeed, simulation environments dealing with fuzzy numbers are implemented with *ad hoc* approaches. Instead, in DEOS a new fuzzy event list is adopted rather than the basic event list. The new *fuzzy aware* event list has been written by reusing code from the previous one. Because of this change, it is not necessary to rewrite the simulator - DEOS modular architecture minimizes such

changes. New fuzzy aware resource and entity classes are derived from the basic ones and new plugins are created in order to collect fuzzy simulation results.
A brief example is reported in order to show the way fuzzy simulation works. A simple production case is considered: a machine has to process two pieces, one piece at a time, depending on the arrival order (FIFO). Machine is initially idle and arrival times for piece 1 and piece 2 are t_A and t_B. In ordinary simulation (absence of fuzzy uncertainty), a feasible event list is showed in Figure 4. First, piece 1 is processed, because $t_A < t_B$. Piece 2 waits in queue (from t_B to t_C) and, when machine gets free at t_C, then piece 2 is processed.

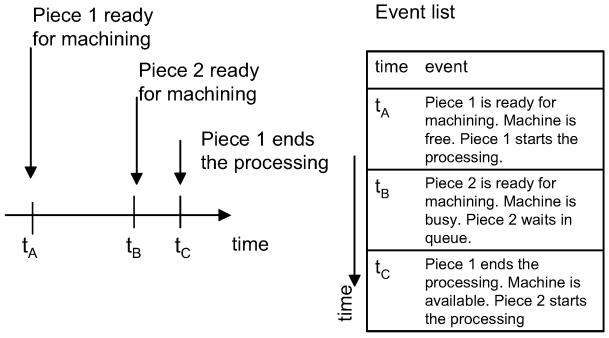


Figure 4: Example of Event List in Ordinary Simulation

If fuzzy uncertainty is present on arrival and processing times, it is not possible to define the corresponding probability distribution. Fuzzy simulation adopts fuzzy sets to model such uncertainty. Fuzzy sets are used instead of crisp numbers (deterministic values or samples extracted from a distribution). A possible situation is represented in Figure 5. Events occurrence time is represented by fuzzy set: not just a single value is expressed, instead a set of values, having different levels of membership, is supplied. For example, the arrival time of piece 1 is represented by fuzzy set π_A that is a set of values *near* t_A - all those values are theoretically possible and no sampling can be performed on the fuzzy set π_A (no statistical distribution is defined). Since no overlapping is present between π_A and π_B, the condition $\pi_A < \pi_B$ is always true - part 1 arrives first at the machine. Then, both conditions $\pi_B < \pi_C$ and $\pi_C < \pi_B$ are possible (π_B and π_C overlap), but $\pi_B < \pi_C$ has greater possibility because of the positions of fuzzy sets. For such a reason, two feasible evolutions can be represented in fuzzy simulation - event list 1 and 2 in Figure 5. In the first, part 2 waits the previous processing end. In the second, part 2 finds the machine free at its arrival (processing of part 1 has already ended).

Each simulation run leads to one of the two feasible evolutions. The *ranking parameter* is changed during the execution of each run (Grieco et al. 2003). Such a parameter permits to have different responses from the ranking algorithm. For example, a possible simulation results summary is showed in Table 5. As it is possible to note, low possibility *event list 2* (replication 1) is less frequent than the other one (replications 2-5): the adopted ranking algorithm is coherent. In DEOS, fuzzy set ranking algorithm can be easily changed by modifying a method of the *event list* class.

Table 5: Fuzzy Simulation Summary

Replication	*Ranking parameter value*	*Ranking condition*	*Event list*
1	0	$\pi_C < \pi_B$	2
2	0.25	$\pi_B < \pi_C$	1
3	0.5	$\pi_B < \pi_C$	1
4	0.75	$\pi_B < \pi_C$	1
5	1	$\pi_B < \pi_C$	1

DEOS and other open source projects

The degree with which the previously considered open source projects can be applied to manufacturing systems is variable. Although all of them provide the basic tools to execute simulation runs, it is possible to note that user interfaces are often neglected and user is forced to learn non-visual languages. Furthermore, the object-orientation is used at programming level, but not always at modeling level. All these lacks raise the amount of skills required to the user and discourage the use of simulation in decision-making contexts. SIMOO is the most appropriate because of its general modeling capability, but no study has been carried out in this direction. OOSIM has been employed in some manufacturing cases, but the resulting work is not open source. DEOS is a suitable open source project especially to deal with manufacturing system modelling. Another valuable DEOS feature is its modular plugin architecture. Different teams of developers can write plugins without the necessity of frequent interactions.

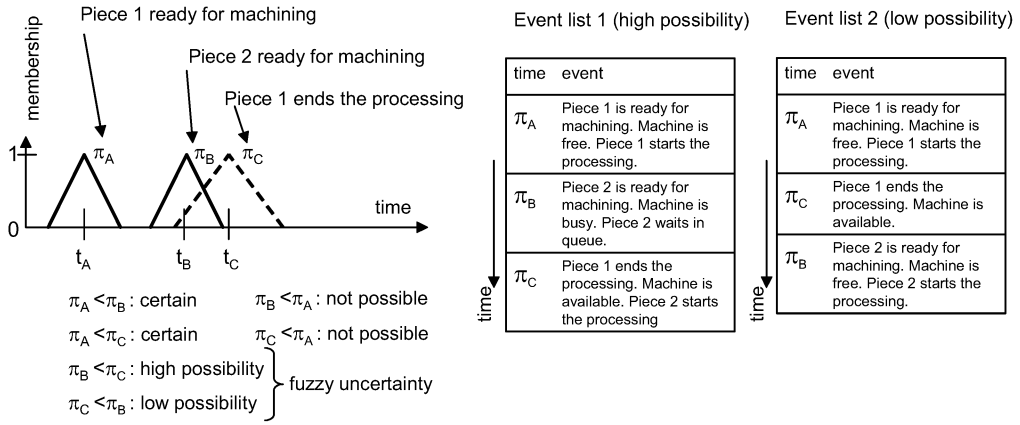


Figure 5: Example of Event List in Fuzzy Simulation

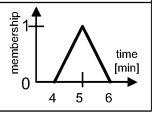
CASE STUDY

In this study a Flexible Manufacturing System (FMS) is considered in order to show DEOS characteristics. The system has automated transporter systems to move pallets (containing parts to be processed) to and from machines, and a L/U station having workers (human operators) that loads and unloads parts from the pallets. Three pallet types are processed, see Table 6 and Table 7. The basic classes have been specialized to simulate objects for different contexts. Inheritance of basic class methods and data allows reusing code. Specialized and basic classes have been used to create the FMS model in DEOS. Moreover, in order to show the potential of DEOS, fuzzy uncertainty has been introduced in L/U station processing times to model human operator working.

Table 6: Case Study Pallet Description

Pallet type	*Number of pallets*	*Number of parts*	*Part type*	*Fixing*
1	2	2	1	1
2	1	4	2	1
3	1	4	2	2

Table 7: Case Study Resource and Time Description

Parameter	*Value*
Number of workers	4
Number of AGVs	4
Number of machines	4
Pallet type 1 processing time	20 min
Pallet type 2 processing time	24 min
Pallet type 3 processing time	12 min
Transport time	1 min
part load/unload time	

The first step to create the FMS model has been to identify the objects participating in the FMS that are eligible to be represented as resources or entities, see Table 8.

Table 8: Resources and Entities in the FMS

Resource types	*Entity types*
L/U station	parts
machines	pallets
workers	
transporters	
pallet buffers	

Next step requires mapping such resource types to the existing ones. In some cases we needed brand new resource types (inherited from the basic classes), but more often we reused the code of the existing plugins. Table 9 shows what we used to simulate the above model components.

Table 9: Reused Code for Modelling FMS Components

Component	*Source code*
LU station	Derived from TDResource basic class
Worker, transporter	Reused code from Server basic resource
Machine	Derived from Server basic resource
Buffer	Mapped to Queue basic resource

In this way, the ordinary FMS model has been created, see Figure 2. Users can place resources on the simulation framework, connect objects and set parameter values - such as processing time for machines, conveying time for transporters and load/unload time for workers - by entering numerical values or statistical distributions. Nevertheless, in order to consider the fuzzy uncertainty (in LU times, see Table 7) an additional step has been made. The fuzzy event list is adopted and a new class of plugins - able to deal with fuzzy numbers - collects the results. The replication number is 101, for each one a specific ranking parameter is adopted from the set {0, 0.01, 0.02, ..., 0.99, 1}.
The *average fuzzy flowtime* for each pallet type is reported in Figure 6. Shape irregularity is due to average operator on fuzzy flowtime values. As it is possible to note the uncertainty in input values leads to uncertain results. Nevertheless, fuzzy simulation permits the study of systems when, for a given parameter, no historical data are available and qualitative information is present only. DEOS allows reproducing both ordinary and fuzzy simulation because of its architecture.

CONCLUSION

An innovative Open Source Object-Oriented Architecture has been presented. In particular, aspects concerning maintainability, result collection and scripting integration are analyzed. The possibility to deal with fuzzy simulation has been exploited. Considering the literature state-of-the-art, innovative aspects concerning DEOS are described.
In future, DEOS will be extended in order to participate in distributed simulations. Special plugins will be designed to allow DEOS to be a federate in a High Level Architecture (HLA) federation and other plugins to wrap DEOS inside an object of the Common Object Request Broker Architecture (CORBA). Moreover, the Tcl question will be further investigated in order to allow the integration of more powerful languages such as Python. Additional fuzzy uncertainty questions will be considered in order to handle uncertain system layouts.

ACKNOWLEDGMENTS

This research has been funded by Italian Minister of Education and Research (MIUR) within the project FIRB titled "Architetture e tecnologie informatiche per lo sviluppo ed evoluzione di software open-source per la simulazione a componenti distribuiti, orientate al settore manifatturiero" (2001-2005 prot. RBNE013SWE PNR 2001-2003, FIRB art. 8, D.M. 199 Ric. 2001).

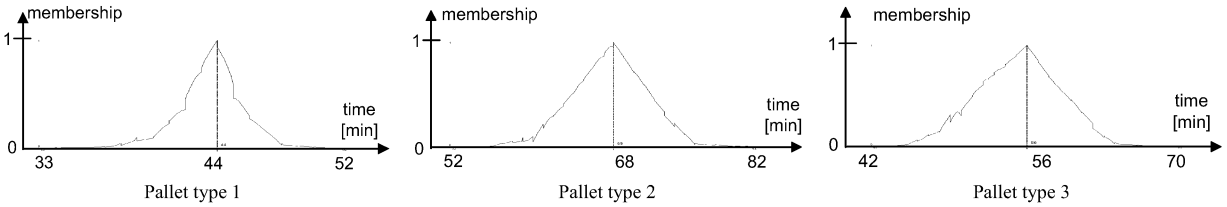


Figure 6: FMS Model Results - Average Fuzzy Flowtime for each Pallet Type

REFERENCES

Arjuna-Team. 1994. "C++SIM User's Guide". Department of Computing Science, University of Newcastle upon Tyne.

Azzaro, C.; Floquet, P.; Pibouleau, L.; Domenech., S. 1997. "A fuzzy approach for performance modeling in batch plant: Application to semiconductor manufacturing". *IEEE Transaction on Fuzzy Systems,* 3 338-357.

Bodner, D.A.; Narayanan, S.; Sreekanth, U.; Govindaraj, T.; McGinnis, L.F.; Mitchell, C.M. 1994. "Analysis of discrete manufacturing systems for developing object-oriented simulation models". *Proceeding of 3rd Industrial Engineering Research Conference* (Atlanta, USA, May). 154-159

Boehm, B. 1981. *Software Engineering Economics*. Prentice-Hall.

Booch. G. 1993. *Object-Oriented Analysis and Design with Applications.* (2nd Edition). Addison-Wesley Pub Co.

Breslau, L.; Estrin, D.; Fall, K. 2000. "Advances in Network Simulation." *IEEE Computer*, 33 59-67.

C++SIM Project documentation: `http://cxxsim.ncl.ac.uk`

CACI 1993. *MODSIM II - The Language for Object-Oriented Programming: Reference Manual*. CACI Products Company.

Copstein, B.; Wagner, F; Pereira, C. 1997. "SIMOO - An Environment for the Object-Oriented Discrete Simulation." *Proceedings of 9th European Simulation Symposium*, 21-25.

DEOS - Open source computer simultion project: `http://deos.unile.it`

Fishwick, P.A. 1991. "Fuzzy simulation: specifying and identifying qualitative models." *Int. Journal of General Systems, 19-3*, 295-316.

Gamma, E.; Helm, R.; Johnson, R.; Vlissides, J. 1994. *Design Patterns: Elements Of Reusable Object-Oriented Software*. Addison Wesley.

Govindaraj, T.; McGinnis, L.F.; Mitchell,C.M.; Bodner, D. A.; Narayanan, S.; Sreekanth, U. 1993. "OOSIM - A Tool for Simulating Modem Manufacturing Systems." *Proceedings of the 1993 National Science Foundation Grantees in Design and Manufacturing Conference*, 1055-1062.

Grieco, A.; Nucci, F.; Anglani, A. 2003. "Representation of fuzzy time variables in discrete event simulation." *Integrated Computer-Aided Engineering*. IOL Pres. Vol. 10, No. 4, 305-318.

Hlupic, V. and Paul, R.J. 1999. "Guidelines for Selection of Manufacturing Simulation Software." *IIE Transaction.* Vol. 31, 21-29.

Liou, T. and Wang. M. 1992. "Ranking fuzzy numbers with integral value." *Fuzzy Sets and Systems,* 50 247-252.

Network Simulator Project documentation: `http://www.isi.edu/nsnam/ns`

Oloufa, A.; Ikeda, M.; Nguyem, T. 1998. "Resource-based simulation libraries for construction." *Automation in construction,* 7 315-326.

OMNet++ Project documentation: `http://www.omnetpp.org`

OOSIM Project documentation: `http://www.isye.gatech.edu/~bodner/research/oosim.html`

Open source scripting language: `http://www.tcl.tk`

SIMOO Project documentation: `http://www.inf.ufrgs.br/gpesquisa/simoo/ index.html`

The Open Source philosophy: `http://www.opensource.org`

Varga, A. 2001. *The OMNeT++ discrete event simulation system. Proceeding of the European Simulation Multiconference 2001*. Prague, Czech Republic.

Yuan Y. 1991. "Criteria for evaluating fuzzy ranking methods." *Fuzzy Sets and Systems,* 44 139-157.

A Heuristic Volume Selection in Dimensional Bounding as an Accelerating Technique in a Ray Tracing

D. Benyamina , N. Hallam
Division of Computer Science and IT
University of Nottingham,Malaysia Campus
Wisma MISC,No 2 Jalan Conlay
50450 Kuala Lumpur
Malaysia

KEYWORDS: Acceleration, Bounding Volumes, Hierarchy organisation, Ray Intersection, Lightning Simulation

ABSTRACT

The most important operation in the ray tracing technique (the most accurate Lightning simulation technique) is to calculate the intersection point between the pixel ray (or the segment launched from the observer eye to the screen pixel) and a set of scene objects, either simple or complex objects (CGS, B_Spline). Optimising this intersection computation, allows substantial time gain in generating the final image. In the goal of optimising the temporal cost needed to generate very complex scenes. An accelerating technique is implemented and shown to be very fructuous in the sense that we have decreased the rendering and visualising time about 60% from the overall time needed to generate the same scene. Implementing the Bounding Volumes method (called also Dimensional Bounding) greatly speeded up the intersection computation which generally represent 75% of the simulation time.

1. INTRODUCTION

The simulation of lightning phenomena is still very expensive when done by an accurate technique such as ray tracing. If we imagine a simple scene comprising a sphere, a plane, and a cube (a total of 12 triangles). For each ray tracing, we have to calculate 14 intersections just to know whether a ray cross an object on its path or not. For a small image with a resolution of 450x320 pixels, there are normally 144 000 rays (in the case of only one ray per pixel), so a total of 2 016 000 intersections[1] to compute without any colour computation (texture) or lighting (ambient, diffused and specular light) and without sending any secondary ray for shading ,reflection, and refraction.

In this paper we will show how the optimisation is performed using the Bounding Volumes method. We will describe a heuristic selection of volume type showing the efficiency of organising data into hierarchy. The other problem arising is how to select a group of object in a volume when constructing the hierarchy.

2. ACCELERATION TECHNIQUES

Different techniques are used to accelerate the computation of rays intersection with environment. They are based on: reducing the cost of ray-environment intersection, minimising the number of ray-object intersections to be tested, minimising the number of rays, and replacing a set of rays that have almost the same origin and the same direction for a common entity.

There are three different and fundamental methods for organising data in order to speed up the computation of ray-object intersection point: Dimensional Bounding (tree) [1], Non-Uniform Subdivision (direct acyclic graph), and Uniform Partial Subdivision methods [4]. Each of these aforementioned methods has an appropriate algorithm to access to a given object included in a volume intersected by a given ray. In this paper we focus on implementing and optimising the Dimensional Bounding method.

2.1 Bounding volumes Method

This methods consists in bounding each environment object by a volume or by a bounding box , with well-known geometric shape, in order to facilitate its intersection with a ray. The same principle is applied on those volumes, until overall environment is included. A complex object can be limited by different parallel planes (polygonal limitations).

This method provides an interesting means to limit the time spent in searching for intersections. In the case where the ray does not cross the bounding volume, the bounded object is not tested and consequently an important amount of computations is discarded.

Even though with the bounding volumes , the order and the frequency within the intersection tests have to be done, are not determined , we can organise the bounding volumes in a hierarchy system.

[1] *Note that an intersection with a plane which is the easiest one, it needs 6 multiplication operations, 5 addition operations and one division.*

2.2 Hierarchies

A hierarchy system is modelled as a tree (direct acyclic graph) as depicted in fig1. Primitive objects are localised in the terminal nodes whereas the bounding volumes are on the intermediary nodes.

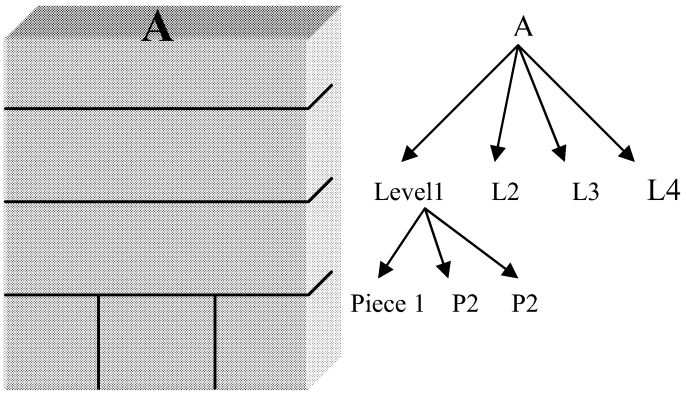


Fig1 : Test the intersection with *pieces 1,2 and 3* only if the projection ray intercept *the block A* and the *level 1*

A low level volume cannot intercept a given ray if its corresponding high level volume does not. Consequently, starting intersection tests at the root (total volume) and going down throughout the hierarchy, every node descendants is (bounding volumes) tested. In each test, if an intersection occurs, its distance will be saved. The volume with the less distance will be considered for the next intersection testing until a leaf node (geometric primitive) is reached.

2.3 How to choose the volume shape?

For an efficient hierarchy organisation, the selection of the volume shape for a given object is very important. The intersection cost is :

$$Cost = n.B + m.I \qquad (1)$$

Where

n: ray number to test for the volume
B: The cost of each test
M: The number of rays that effectively reach the volume
I: Object intersection cost

Note that *n* and *I* are fixed.

Minimising *B* is choosing the less expensive volume in term of time of intersection calculation, while minimising *m* is choosing the narrowest volume.

An object can be limited by a volume which can have any form: normal, oriented, or distorted. Note that it can also be limited by associated volumes (using Union, Intersection. etc.) of those volumes. In this case, the ray has to check the intersection with overall volumes before testing intersection with the object itself.

To have the best volume that fits very well the bounded object, many properties are to be taken into consideration. We focus on the properties of rays` distribution to test on that object. In other word, to study the volume effect on the distribution of rays observed by one or many volumes inside it. Goldsmith&Salmon [1], in their approach, have calculated a conditional probability: the probability that an arbitrary ray **r** reach a volume **B** knowing that **r** reach the volume **A**, fig 2. We consider that the rays` distribution on A is uniform, which is not true in practice.

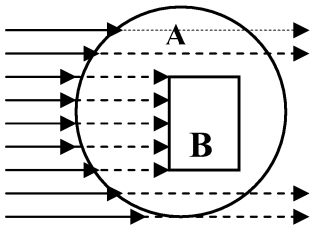


Fig 2 : The probability that a ray reach **B** knowing that it reach A

So we have

$$Pr(r \text{ reach } B / r \text{ reach } A) = \langle P(B,d)\rangle / \langle P(A,d)\rangle = S(B)/S(A) \qquad (2)$$

Where :

P(V,d) : projection air of volume V for a distance d
< > : The average taken in all directions
S(V) : V surface

The intersection cost (volume, ray) is divided into two components:

External cost (EC): A fixed cost of testing intersection with the volume itself, corresponding to *B* in eq.1.
Internal Cost (IC): Average intersection cost of a ray with the content of that volume, corresponding to *I* in eq.1.

In general, for a volume *A* enclosing several volumes *B1,B2,...,Bn* we have :

$$IC(A) = \sum \{ EC(Bi) + S(Bi)/S(A) \,.\, IC(Bi) \}. \qquad (3)$$

IC = 0 for a primitive object.

We can see from eq.3 that the average cost of intersection of a given ray with a volume in the hierarchy, is expressed in terms of *EC*, and surface air. Thus the volume cost depends on the shape of that volume and on volumes and objects it includes.

3. Optimisation using Bounding volumes

We have chosen spheres and parallelepipeds as bounding volumes, because of the cost in term of time-machine which is the less expensive compared to any other volume. Firstly, we take sphere as a bounding volume to clarify the optimisation method implemented.

We defined two classes:

1. class Sphere: The sphere comprising the objects

```
Class Sphere  {  Point3d centre;
                 double  R,  R2;
                 Surface   *surf;
               };
```

2. class Bounding_Sphere: The sphere that includes the whole scene, (all objects).

```
class Bounding_Sphere
{
 Sphere      *sph;
Tlist        * list_sphere;
Tlist        * list_surface;
};
```

For each object in the scene, we have to determine its bounding sphere which is different from one object to the other, and then appending it in the list *list_sphere* (class *Bounding_sphere*). The infinite objects, such as planes, are inserted in the list *list_surface*.

For each object, two values are needed to define its bounding sphere: the maximal and the minimal values (*max, min*) on the axes in three dimensions. The sphere centre is the point located at equal distance from *max* and *min* therefore the radius can be deduced (the half of this distance).

The process of optimisation is as follows. First, the intersection point, *t*, between of a ray with a scene is determined. The one that is the closest to the ray with the surfaces of the list *list_surf* is sought for.
Then another test is made to check whether there is an intersection between the ray and the bounding volume of the scene. If no such intersection, then the closest point is set to the point *t* calculated previously.

If such intersection exists, then each of the spheres in *list_sphere* is tested to see whether there is intersection with the ray, in which case, the new point *t*, as a result of the intersection of the ray with the object bounded by the sphere.

4. Optimisation Based on the Hierarchy Organisation

The optimisation method used above does not take into consideration the objects position in the scene. To ameliorate this function, we utilise the hierarchy principle in intersection test. To build the hierarchy tree, we involve only the objects bounding volumes without creating other volumes. For a given object volume, we check if it is bounded by another volume in the scene and is added to the list *list_bounding_sphere*, and the same operation is applied to overall volumes in the scene.

```
Class Sphere_hierarchy
{
public :
    Sphere   *sphere;
     Tlist   *list_bounding_sphere:   // contains volumes of
                                      //type bounding_sph
    Tlist  *list_surface;
    Surface  *surface
    Bool  include(const Sphere  *sph)  const; // check if a
                          //sphere include another one

    double inter(Ray  *ray); // is a recursive function
         //which check if there is intersection with a
         //bounding_sph. In that case another test is
         //invoked, to check /intersection with all
         //bounding_sph in its list list_bounding_sphere

    ~Sph_include_sph();
 };
```

5. Optimisation based on the initial computation

In addition to the optimisation method in the previous section, some good programming hints are followed in order to get maximum efficiency. For a simple still image with a resolution of 800x600, 480 000 principal rays are lunched and about 4.8 millions of secondary rays. Each saving in time that is spent in computing the pixel colour is fructuous, even if it incurs the addition of complication and complexity of the program code. Thus we avoid to repeat the same calculation 5 millions times (About 20 seconds of machine computation). According to the results obtained, the gain is impressive, and divides in half the image tracing time. The constant expressions such as Ray-source coordinates, lighting source position, could be calculated early, since they do not change during the image calculation process.

6. RESULTS

The results of the examples shown below, were executed on a very slow machine, which is a Pc – Pentium1, a frequency of 333MHZ, graphic card with 16 millions of colors and 32 MO of RAM. For a simple scene comprising 29 objects and illuminated by two luminance sources, for an image resolution of 350x220, (Figure 1, table1).

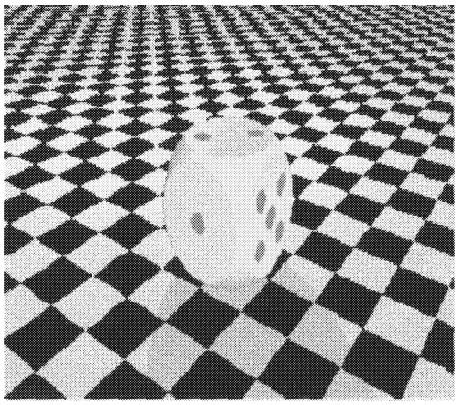


Figure1: First simulated scene

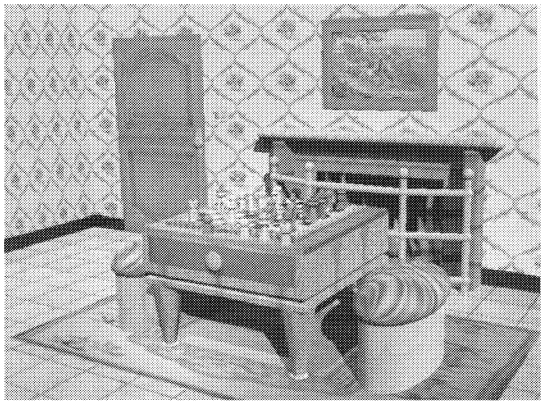


Figure2: second simulated scene

Number of :	
surfaces :	29
luminous source :	2
primary rays:	77000
shadow rays:	147485
reflection rays:	0
refraction rays:	0
Total number of rays: **224485**	

Table1:statistics for figure1

Number of :	
surfaces :	166
luminous source :	2
primary rays:	231000
shadow rays:	545514
reflection rays:	51417
refraction rays:	0
Total number of rays:**2827931**	

Table2: Statistics for figure2

We obtained the following data according to the optimisation method applied:

1. Using simple optimisation (based on initial computations)
   Time passed (sec) : 30.38
2. Optimisation using bounding spheres
   Time passed (sec): 14.76
3. Optimisation using bounding spheres organized in hierarchy
   Time passed (sec):13.466

- Another attempt was made for a more complex scene with 166 objects, 2 luminance sources with image resolution of 550x420, (figure 2, table 2).

We obtained the following data according to the optimisation method applied :

- Using simple optimisation (based on initial computations)
  Time passed (sec) : 580.269
- Optimisation using bounding spheres
  Time passed (sec): 271.61
- Optimisation using bounding spheres organized in hierarchy
  Time passed (sec): 216.881

CONCLUSION

A ray-tracing technique is still an efficient tool in visual simulation to create realistic images. Its power comes from the fact that the realisation of a ray tracer is incremental. We can exploit the coherence at some stages in the image generating process, where the 'ray coherence' [3] and the 'surface coherence' can be implemented. We can adapt our algorithm to the hierarchy methods and their new adaptive bases introduced in [2][5], it reduces the cost and the complexity of synthesis image algorithms but their main problem is to generate a discrete representation which satisfy shadow representation [6,7].

REFERENCES

[1] James Arvo and David Kirk
"A survey of ray Tracing acceleration techniques ray-tracing", (220-259), 1988.

[2] A.T. Campbell, D. Fussel, "Adaptive Mesh Generation for Global Diffuse Illumination", Computer Graphics, 24(4), (155-164), 1990.

[3] Cohen-Or, D., E. Rich, U. Lerner, and V.Shenkar, "A Real-Time Photo-Realistic Visual Flythrough," *IEEE Transactions on Visualisation and Computer Graphics,* 2(3), (255-265), September 1996.

[4] J.Foley, A.Van Dam, S.Feiner, J.Hughes, R.Phillips, "Introduction a l`infographie" , Edison Wesley, 1995.

[5] P. Hanrahan, D. Salzman, L. Aupperle, “A Rapid Hierarchical Radiosity Algorithm”, Computer Graphics, 25(4), (197-206), 1991.

[6] D. Lischinski, F. Tampieri D.P. Greenberg, “Discontinuity Meshing for Accurate Radiosity”, IEEE Computer Graphics, 12(6), (25-39), 1992.

[7] D. Lischinski, F. Tampieri D.P. Greenberg, “Combining Hierarchical Radiosity and Discontinuty Meshing”, Computer Graphics Proceeding, (199-208), 1993.

[8] J.R. Wallace, K.A. Elmquist, E.A. Haines, “A ray-tracing Algorithm for Progressive Radiosity”. Computer Graphics, 23(3), (315-324), 1989.

STATIC AND DYNAMIC EQUAL OPPORTUNITY EVALUATION

Ignacio J. Benítez
José L. Díez
Juan A. Lacort
Pedro Albertos
Department ISA
Polytechnic University of Valencia
Camino de Vera, s/n 46022 Valencia, Spain
E-mail: pedro@isa.upv.es

KEYWORDS
Simulation Tool, Diversity, Decision Support Systems, Social Science.

ABSTRACT

Achieving diversity at the workplace is a difficult task and eradication of diversity constraints is a complex assignment to deal with. The main goal of this paper is to present a systems engineering view of the existing diversity within organizations. This objective is accomplished by depicting a diversity model where inputs, outputs, interferences and control actions are clearly identified. In order to allow equal opportunity enhancement in organisations, a model-based methodology for static and dynamic analysis is proposed and a software tool presented.

INTRODUCTION

Thinking about the staff of an organization in terms of a valuable resource is getting more and more common. At the same time, taking advantage of the diversity of the members of an organization has proven to be a well rewarded strategy (Stockdale and Crosby 2004). The diversity concept can be applied to different characteristics, such as gender, age, race, nationality, etc. Equal Opportunities (EO) measures in the organizations through the application of the diversity criteria allow pointing out the organization's performance in using its human resources. It is a matter of emphasizing meritocracy, above origin or condition.

The University environment has been an attractive scenario to study about the gender EO, mainly because of the University complex structure, made up of different activities and levels of operation, (alumni, faculty, assistants…).

The EO issue has been studied in some European projects. One of the best known is the research project "Equal opportunities at Universities. Towards a gender mainstreaming approach" (Stevens and Van Lamoen 2001). The main aim of the project was to develop a manual for gender EO based on the mainstreaming strategy, as a result of analyzing the information and experiences concerning equal opportunities policies and gender mainstreaming in universities and research institutions.

Further to this project, the main purpose of our study is to analyze the EO condition of an institution by means of the automated process of a set of heterogeneous data, and the evolution of diversity and EO through time.

The main obstacle to analyze systems behaviour is based on the qualitative and subjective nature of the information. Structures and relationships, just to cite some data, present the same weakness (Albertos and Barberá 1996), complicating the study and requiring advanced techniques. There have been interesting approaches to deal with this characteristic (Helbing 1995) and fuzzy logic has been proposed (Barberá and Albertos 1994) as a mathematical tool to represent the approximated knowledge implicit in qualitative measurements.

A further step in the analysis of an organization is to predict how the conditions will vary in the future. In order to achieve that, a dynamic analysis has to be done.

A number of EO indices, most of them of a qualitative nature, as well as a set of corrective actions have been proposed in the framework of the European Project "Divers@: Gender and Diversity" the authors are involved in. This research is partly funded by the European Social Fund (Project ES296).

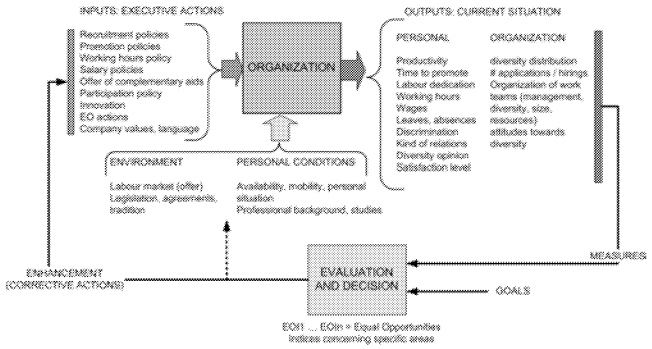
One of the goals of the project is to develop a methodology to revise and analyze the presence of the EO indices and the role of diversity and gender perspective in these organizations, suggest corrective actions when necessary and extend the study over time, controlling the evolution of the process.

Further research is presented in this paper. It is organized as follows: in the next section a process view of diversity in organizations is presented. Then a model for diversity analysis is detailed, split in static and dynamic analysis in organizations. A software tool developed with the aim of helping experts in the static analysis is also outlined. A final section listing some conclusions is included.

THE DIVERSITY AS A PROCESS

For the sake of an accurate analysis and detection of the situation of diversity and EO within an organization, a complete feedback system has been designed, as an analogy with a classical control systems configuration (Ogata 2001). This system is depicted in Figure 1.

The diversity is seen as a variable, whose value needs to be optimized. The variable diversity flows through the process called "organization", which has output signals that are function of internal variables, inputs and interferences, all of them related to diversity.



Figures 1: the Diversity as a Process

The inputs are the executive actions, all the aspects affecting diversity and EO that can be changed at the organization's level, such as offer, promotion or working hours policies. The interferences comprehend the variables which affect diversity and EO but can not be directly addressed by the organization; the diversity at the offer or labour market, for example, and personal conditions such as values, physical conditions and mobility or family obligations would be interferences to the process. The outputs are variables considered relevant or valuable to measure diversity and EO. Examples of variables that can be assessed are the satisfaction level among workers, differences in wages or the diversity distribution at the different levels of the organization.

This information can be gathered in different ways, through interviews, questionnaires, and organization's documents. After an adequate pre-processing of data, an appropriate analysis of the information will modify values on a set of pre-defined EO indices. An evaluation is made and specific actions are suggested with the aim to enhance the current situation; these actions are input variables to the organization. It is expected that, when suggested actions are effectively applied, an improvement in diversity and EO will be obtained in time.

The observability of the system and its enhancement are not continuous, but discrete in time. Due to the nature of the data being measured, which is slow to collect and needs to be pre-processed, an analysis of diversity and EO can be done only from time to time, and not continuously. Moreover, changes in diversity within an organization due to suggested actions and awareness are not immediate, so there is a latency until results are visible. Therefore, whenever the measures are available, a *static* analysis is done, taking the data, evaluating them and producing a result. But another further, overall historical analysis becomes necessary, to study the *dynamic* of the system and to estimate future trends for diversity and EO at an organization. Both static and dynamic analyses are detailed next.

STATIC ANALYSIS

Any time the data is available, an analysis can be done: the analysis will evaluate current data and give a diagnosis of the current situation, suggesting corrective actions. This is the static analysis, and it follows two steps: a pre-processing of the available data, to gather the information in the adequate configuration, and the analysis of that pre-processed information.

Data Acquisition and Pre-processing: Clustering

The information must be arranged in areas of special interest for the analysis of diversity and the assignment of values to certain EO indices (called *constraints* indices) stated by experts. It must be emphasized that this information is obtained in different ways and can be of a quantitative and a qualitative nature (Helbing 1995). The pre-processing of this information tries to capture the global current situation of diversity and EO at that given time within the organization being analyzed.

The proposed method to arrange the information and extract global EO constraints indices is by means of fuzzy clustering techniques. Classical clustering methods classify objects or individuals in clusters, according to a similarity, in a mathematical sense (Bezdek 1987). In its simplest approach, an object belongs to one cluster exclusively. Given a set of N objects $X = \{x_1, ..., x_N\}$, each object x_k having a number j of characteristics or dimensions, $x_k = \{x_{k1}, x_{k2}, ..., x_{kj}\}$, the classical partitioning space M_c in an expected number of c clusters is defined by a partition matrix $U \in R^{cxN}$ whose elements μ_{ik} (membership factor for cluster i of object k) are defined by Equations (1), (2) and (3).

$$\mu_{ik} \in \{0,1\}, \quad 1 \le i \le c \quad 1 \le k \le N \qquad (1)$$

$$\sum_{i=1}^{c} \mu_{ik} = 1, \quad 1 \le k \le N \qquad (2)$$

$$0 < \sum_{k=1}^{N} \mu_{ik} < N, \quad 1 \le i \le c \qquad (3)$$

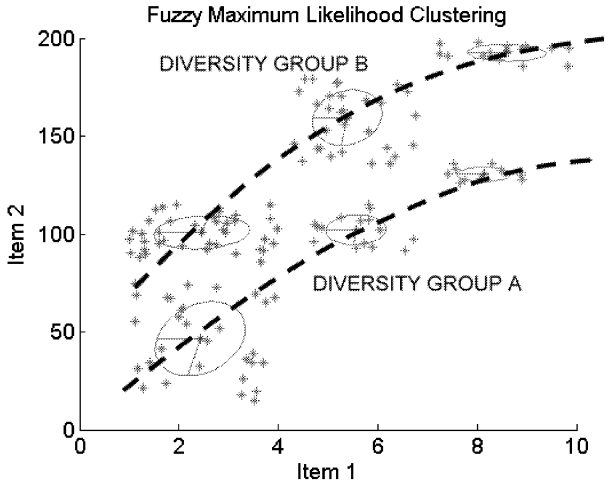
Undefined situations may appear when an object does not clearly fall within a cluster, but near two or more of them, or when two or more clusters overlap.

A much more realistic analysis can be done using fuzzy clustering algorithms (Bezdek 1987). These algorithms extend the classical approach providing more flexibility. Objects being fuzzy clustered do not belong to a specific cluster exclusively, but to them all at different percentages. Thus, every object k has a different membership factor μ_{ik} for every cluster i, being the sum of all percentages again equal to 100%. The only difference, then, between classical M_c and fuzzy F_c partitioning spaces, is that the membership factor μ_i in M_c takes values 0 or 1, while μ_i in F_c varies from 0 to 1. Thus, Equation (1) becomes:

$$\mu_{ik} \in [0,1], \quad 1 \le i \le c \quad 1 \le k \le N \qquad (4)$$

Applying a fuzzy clustering algorithm, such as the fuzzy c-means (Bezdek 1987) or the fuzzy maximum likelihood

(Dunn 1974), to the arranged available outputs measured by the experts will produce as a result two matrices. One contains the mathematical centres of the clusters (matrix of centres), having as many centres as clusters have been demanded, all with the same dimensions, one for each characteristic analyzed. The other matrix (fuzzy partition matrix) indicates, for each object or item, the membership percentage (varying from 0 to 1) to each cluster. Thus, the elements of the fuzzy partition matrix represent scaled membership values of every item being analyzed (column) to every cluster (row). A graphical example of fuzzy clustering can be seen in Figure 2, and the numerical results (matrices) produced are shown in Equations (5), (6), (7) and (8).



Figures 2: Example of Fuzzy Clustering

$$centersB = \begin{pmatrix} 2.4536 & 100.8894 \\ 5.3857 & 159.5584 \\ 8.5934 & 192.5582 \end{pmatrix} \quad (5)$$

$$membershipB = \begin{pmatrix} 0.9988 & 0.9971 & 0.9980 & \dots \\ 0.0012 & 0.0028 & 0.0020 & \dots \\ 0.0000 & 0.0001 & 0.0000 & \dots \end{pmatrix} \quad (6)$$

$$centersA = \begin{pmatrix} 2.4256 & 46.6160 \\ 5.5575 & 102.1263 \\ 8.1350 & 130.5033 \end{pmatrix} \quad (7)$$

$$membershipA = \begin{pmatrix} 0.9966 & 0.9983 & 0.9988 & \dots \\ 0.0032 & 0.0016 & 0.0011 & \dots \\ 0.0002 & 0.0001 & 0.0001 & \dots \end{pmatrix} \quad (8)$$

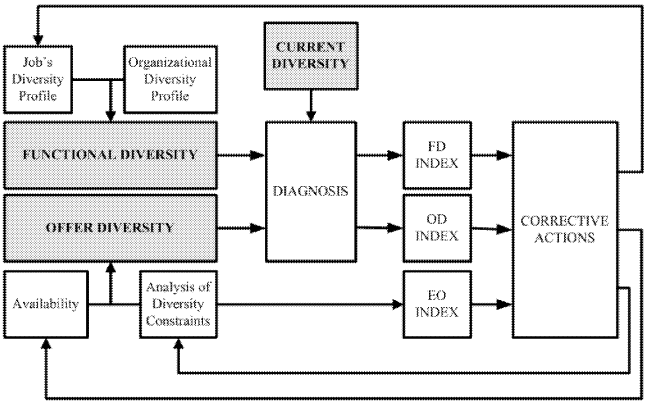
The example in Figure 2 shows the result of applying fuzzy clustering on two different diversity groups (A and B), having both groups two related items of analysis (two dimensions). In this example, three different clusters with their centres have been searched for each group. The centres are displayed as matrices with two dimensions in Equations (5) and (7). The fuzzy partition matrices are also shown (the first three columns only) in Equations (6) and (8), so it can be seen that each object (column) has three membership values, being the summation of the three values equal to one.

Fuzzy clustering algorithms are a powerful tool to obtain EO indices concerning specific related topics of interest stated by the experts, such as a relation, for instance, between productivity and discrimination perception for each diversity group.

Once the information has been conveniently arranged and the values for the desired targeted indices have been produced, an analysis is done to generate a diagnosis of the situation and to suggest corrective actions.

Data Analysis: Evaluation and Results

An analysis of diversity is done, based on a model that relies on three different perspectives (functions) of diversity: the *functional* diversity (D_f), the *offer* diversity (D_o) and the *current* diversity (D_c). The model of evaluation is depicted in Figure 3.



Figures 3: the Diversity Model

The functional diversity (Benítez et al. 2003b) is a profile of diversity specifically designed for the required job. It is based in the study of the characteristics needed for the position (Kite 2001), and the rates of diversity inside and target of the organization (the so called *internal* and external *client*).

The offer diversity (Benítez et al. 2003b) is the result of the available diversity at the labour market, taking into account the detected diversity constraints. It is the workplace constraints (C) what mainly determines short term variations in the offer diversity. A comparison is made between functional and offer diversity and a profile of achievable diversity is obtained, which is evaluated along with the current diversity at the work team or department being analyzed. The result is a *diagnosis* of the situation of diversity at that moment, expressed numerically in terms of three indices: functional (FD), offer (OD) and equal opportunities (EO).

Along with a knowledge base provided by experts, *actions* can be suggested, to be applied on different areas and with different priorities. These actions are built and selected by the use of *intelligent* techniques, such as fuzzy logic (Wang 1997).

As stated before, if corrective actions are applied, an enhancement in diversity and EO is expected to be obtained in a period of time.

DYNAMIC ANALYSIS

In the particular case of diversity study in organizations, dynamic analysis could be very interesting, provided the information about the future effect produced by some corrective actions is given.

Diversity model in Figure 3 shows that corrective actions would provide a change in the diagnosis indices through variations at Functional Diversity and Offer Diversity. These changes lie on, and would be detected by, changes in the data collected from the organization: Job's Diversity Profile, Organizational Diversity Profile, Availability and Diversity Constraints. As Figure 3 shows, Organizational Diversity Profile can not be modified at all. Although a change in Job's Diversity Profile could be performed by redefining the position under study, it would have immediate effect on D_f and lead to a different analysis rather than to a dynamic study of the previous scenario. Additionally, the Job's Diversity Profile is usually considered to be fixed if the position under study is not changed and a redefinition of the characteristics for the position is not allowed. Then, D_f can be considered a constant and the dynamic effect in the system would be mainly due to D_o.

Dynamic model $D_o(t)$ is a combination of the effects of Availability and Diversity Constraints. It should be emphasized that although corrective actions for improving Availability can be previously defined, and some of them could be applicable at the organizations level, most actions (especially those with important effects on labour market) are related to policies to be implemented by governments (law, assistance, …) and therefore are outside the scope of the organization. Actually, most corrective actions are focused on diversity constraints (C). Then, if previous simplifications apply, C is the main responsible for the dynamics of the system, and the main influence in $D_o(t)$ variation, as expressed in Equation (9).

$$C(t) \rightarrow D_o(t) \Rightarrow \Delta D_0 = f(\Delta C) \qquad (9)$$

Therefore, the problem of dynamic analysis of diagnosis indices evolution is reduced to variations on C. Let $C(t_o)$ be the numerical value obtained from an initial study of diversity constraints. In order to forecast the value $C(t_i)$ for diversity constraints in the sampling time *i*, the decrease in diversity constraints $\Delta_{oi} = C(t_o) - C(t_i)$ must be estimated.

Estimation of Δ_{oi} appears to be difficult. However, the usual procedure followed by experts in a diversity constraints analysis is to inquiry a group of workers. The test taken by workers is composed by different questions, each of them related to a specific constraint. Although some human characteristics are time varying and their evolution can be strongly influenced by external actions as well as working conditions (Albertos et al. 2003; Benítez et al. 2003a), the experts can devise the typical effect that the application of a corrective action would produce in the answer of a question taken at a specific time after its application. Consequently, dynamic model of the organization is a relation (provided by experts) of the *increases in value* that each corrective action would provide in the inquiry answers and *when* this effect would be appreciated (partially or totally).

Then, corrective actions lead to effects on diversity constraints with the following characteristics to be defined: when a corrective action will show an initial effect in the sense of variations in inquiry answers (delay), when a corrective action will show its total effect in the sense of variations in inquiry answers (rise time), and valuations by workers at each two previously defined times. Unless other specification is done by the expert, a linear evolution can be supposed and simulations can be done.

THE DIVERS@T SOFTWARE TOOL

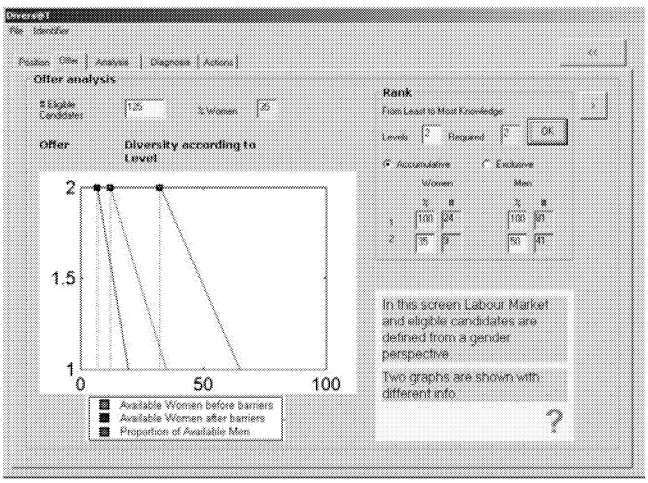
Having defined this methodology of static analysis, a software tool called Divers@T has been developed to study gender diversity and EO at work teams. The Divers@T application comes as part of the project "Divers@: Gender and Diversity", the authors are involved in.

As main, reliable features, Divers@T computes the information from a desired position at a workplace, the EO indices gathered in form of questionnaires among workers, and the data from the available offer, and produces both Functional and Offer Diversities suggesting ranges of optimal selection; an evaluation of gender diversity and EO after a selection has been made, and a complete numerical and lettered diagnosis along with a list of suggested corrective actions, classified in levels of priority and term.

Previous versions of this software have been described in detail in other authors' works (Benítez et al. 2003b; Lacort et al. 2004).

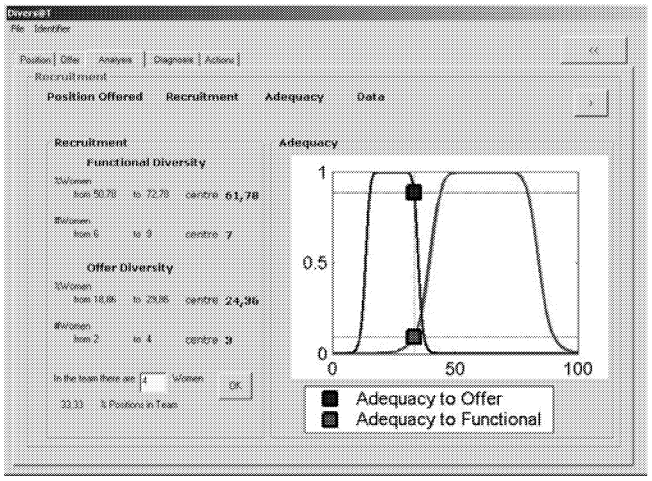
Nevertheless, an illustrated example is included next, to clarify the operation of static analysis of diversity and EO at the workplace.

Figure 4 depicts the labor market distribution, attending to gender. A further classification is done based on the skills provided by the candidates. The three lines inset in the graph represent from right to left and in percentage by level, available men, available women in absence of gender barriers and current available women.



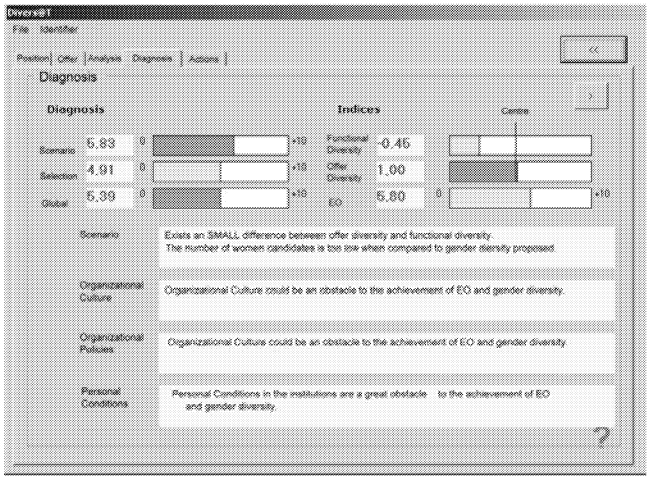
Figures 4: Obtaining the Offer Diversity on Divers@T

Once the labor market has been analyzed, the size of the team is established and a proposal of diversity distribution is done. The adequacy of the final decision in the selection is presented both compared to Offer and to Functionality. The adequacy index is obtained by projecting results to the vertical axis. Figure 5 shows an example.



Figures 5: Offer and Functional Diversity Analysis on Divers@T

Results are evaluated and a complete diagnosis is produced. As can be seen in Figure 6, the numerical indices for Functional and Offer Diversities and EO are shown in the top right. In the top left, a numerical diagnosis is produced. Diagnosis sentences are displayed at the bottom.



Figures 6: Diagnosis and Evaluation on Divers@T

A comprehensive list of corrective actions is also produced, classified in areas of application, relevance and term.

ACKNOWLEDGEMENT

We would like to thank the research group from the Institute of Women Studies (IUED), at the University of Valencia, Spain. Led by Ester Barberá, they have been the promoters of the project Divers@: Gender and Diversity and the essential source of an adequate knowledge base for the properly operation of the Divers@T software tool.

CONCLUSION

A methodology for the analysis and evaluation of EO and diversity at the workplace has been presented. This analysis has been clearly divided into static and dynamic evaluation, being the two sections fully explained.

Furthermore, the static analysis methodology has been put into practice with the development of the Divers@T software tool, outlined in this paper and referenced in other works.

Future developments and research include the implementation of dynamic analysis in Divers@T and the design of a different software application aimed to analyze any diversity kind, such as race, creed, age, etc. and the EO issues that could concern it.

REFERENCES

Albertos, P. and Barberá, E. 1996. “Control Structures in Motivational Processes”. In *Proceedings of the XIII IFAC World Congress. S. Francisco*. Vol. J, 101-106.

Albertos, P., Benítez, I. J., Díez, J. L. and Lacort, J. A. 2003. “Tool for equal opportunity evaluation in dynamic organizations”. In *Proceedings of 13th IFAC Symposium on System Identification*. Elsevier Science Ltd, Oxford, UK.

Barberá, E. and Albertos, P. 1994. Fuzzy Logic Modeling of Social Behavior. *Cybernetics and Systems*. Vol 25, No.2, 343-358.

Benítez, I. J., Albertos, P., Barberá, E., Díez, J. L. and Sarrió, M. 2003. "Equal Opportunities Analysis in the University: The Gender Perspective". In *Eurocast 2003. Cast and Tools for Complexity in Biological, Physical and Engineering Systems*. ISBN: 84-688-0820-2. REGISTER: 341403, 96-98.

Benítez, I. J., Díez, J. L., Lacort, J. A., Albertos, P. and Candela, C. 2003. “Simulation Tool To Analyse Gender Diversity At Workplace”. *ESMc'2003*. ISBN 90-77381-04-X EUROSIS - ETI, 92-96. Ghent, Belgium.

Bezdek, J. C. 1987. *Pattern recognition with Fuzzy Objective Function Algorithms*. Plenum Press.

Dunn, J. 1974. "A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters". *Journal of Cybernetics*, Vol. 3, 32-57.

Helbing, D. 1995. *Quantitative Sociodynamics*. Kluwer Academic Publishers.

Kite, M. 2001. "Gender stereotypes. An Encyclopedy of women and gender. Sex similarities and differences, and the impact of society on gender". Academic Press , Volume 1 (A-K), 561-570.

Lacort, J. A., Díez, J. L., Benítez, I. J. and Albertos, P. 2004. "Divers@T: a Tool to Study Gender Diversity in Organizations". *SIMULATIONS: Transactions of the Society for Modelling and Simulation International*, unpublished.

Ogata, K. 2001. *Modern Control Engineering* (4th Edition), Pearson Education. ISBN: 0130609072.

Stevens, I. and Van Lamoen, I. 2001. *Manual on Gender Mainstreaming at Universities*. Garant, Leuven - Apeldoorn.

Stockdale, M. S. and Crosby, F. J. 2004. *The Psychology and Management of Workplace Diversity*, Blackwell Publishing, Oxford, UK.

Wang, L.-X. 1997. *A Course in Fuzzy Systems and Control*. Prentice-Hall. New Jersey.

Discovery of New Global Minima for Lennard – Jones Atomic Clusters Using TRUST Simulations

Jacob Barhen [1,*] Vladimir Protopopescu [1] Gabriel Amselem [1, 2] Julien Dahan [1, 2]
Christopher Randall [1, 3] Zachary Vane [1, 4]

[1] *Center for Engineering Science Advanced Research*
Oak Ridge National Laboratory
Oak Ridge, TN 37831, USA

[2] *École Supérieure de Physique et Chimie Industrielles*
75005 Paris, France

[3] *Alabama A&M University*
Normal, AL 35762, USA

[4] *Virginia Polytechnic Institute*
Blacksburg, VA 24061, USA

Key Words: global optimization, Lennard-Jones clusters, Visual FORTRAN95 simulation, TRUST

Abstract. The preeminent importance of nano-scale materials science is driving the growing interest in discovering the conformation of atomic clusters with lowest energies. Despite its simple formulation, this modeling and simulation problem is extremely hard to solve. The TRUST algorithm, one of the most powerful global optimization (GO) approaches available to date, is applied here for the first time, to determining the conformation of *Lennard-Jones* clusters. TRUST identifies lower minima than reported with previous methods and reaches them at considerably lower computational cost.

1. Introduction

In recent years, there has been a growing interest in modeling and simulation of clusters, i.e., of aggregates of atoms or molecules [Wales and Scheraga 1999, Cai *et al* 2002, Locatelli *et al* 2003, Johnston 2003, Lee *et al* 2003]. This interest is driven by the realization that the material properties of clusters may be very different from those of discrete molecules or bulk matter. The primary objective of cluster simulations is to determine the atomic (or molecular, or ionic) spatial conformation of the lowest potential energy. Bonding models used to describe the interaction between the particles in the cluster may range from simple phenomenological potentials to those derived from *ab initio* molecular orbital calculations or density functional theory.

In this study, we focus on atomic clusters where the interaction is described by the Lennard-Jones (LJ) potential. More precisely, the function to be minimized is the total potential energy of an N atom cluster, given by:

$$E = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left(\frac{1}{r_{ij}^{12}} - \frac{2}{r_{ij}^{6}} \right), \qquad (1)$$

where r_{ij} is the distance between atoms i and j in the cluster. Several deterministic and stochastic methods applied to optimizing the LJ clusters have been reported [see e.g., Hoare and Pal 1971, Hoare and Pal 1972, Northby 1987, Xue 1994, Deaven *et al* 1996, Leary 1997, Wales and Doye 1997, Wolf and Landman 1998, Leary 2000, Cai *et al* 2002, Locatelli *et al* 2003, Lee *et al* 2003]. These methods have been assessed by various measures of performance such as: (*i*) the number of function iterations needed to attain the (putative) global minimum; (*ii*) the value of the attained minimum; (*iii*) the robustness of the method w.r.t. the initial configuration and seed; (*iv*) dealing with challenging situations (e.g., for N = 75, 98, 102); (*v*) percentage of successful cases; etc.

It is interesting to note that, over the years, even minute (less than fractions of a percent) improvements in the value of the global minimum have been considered sufficiently relevant to be reported [e.g., Hoare and Pal 1971, Xue 1994, Wales and Doye 1997]. This should indeed be the case, since the GO problem for the LJ clusters is extremely hard. The number of local minima grows at least like $O(e^N)$, which makes the energy surface a very difficult landscape to explore and sort out.

The remainder of this paper is organized as follows. The TRUST method is briefly reviewed in Section 2. In Section 3, we describe the new algorithmic and code developments, and in Section 4 we present and discuss our results and outline future research directions.

2. The TRUST Method

TRUST is a method for solving continuous GO problems [Barhen *et al*, 1997] that has not only achieved leading edge performance on the standard SIAM benchmarks, but has also shown promising capability for large-scale exploratory seismology problems [Reister *et al* 2001]. It builds on two innovative concepts, namely subenergy tunneling and non-Lipschitzian terminal repellers, to ensure escape from local minima in a fast and computationally efficient manner.

* Corresponding author: barhenj@ornl.gov

The generic GO problem solved by TRUST can be stated as follows. Determine

$$\mathbf{x}_g = \arg \min_{\mathbf{x}\in\mathcal{D}}\{f(\mathbf{x})\}, \quad \text{where } f:\mathcal{D}\subset\mathbb{R}^{3N}\rightarrow\mathbb{R}. \tag{2}$$

For the considered application, the objective function f is the conformational (potential) model energy, and $\mathcal{D}$ is the domain of the model parameters $\mathbf{x}$, a 3*N*-dimensional vector representing the spatial coordinates of the *N* particles in the cluster. The result of the optimization process produces the optimal (minimum energy) configuration, $\mathbf{x}_g$, and the corresponding global minimum value of the objective function, $f_g = f(\mathbf{x}_g)$.

The filter effect arises from the observation of the following approximation:

$$E_{sub}(x,x^*) \cong \begin{cases} 0 & \text{if } f(x) \geq f(x^*) \\ f(x) - f(x^*) & \text{if } f(x) < f(x^*) \end{cases} \tag{6}$$

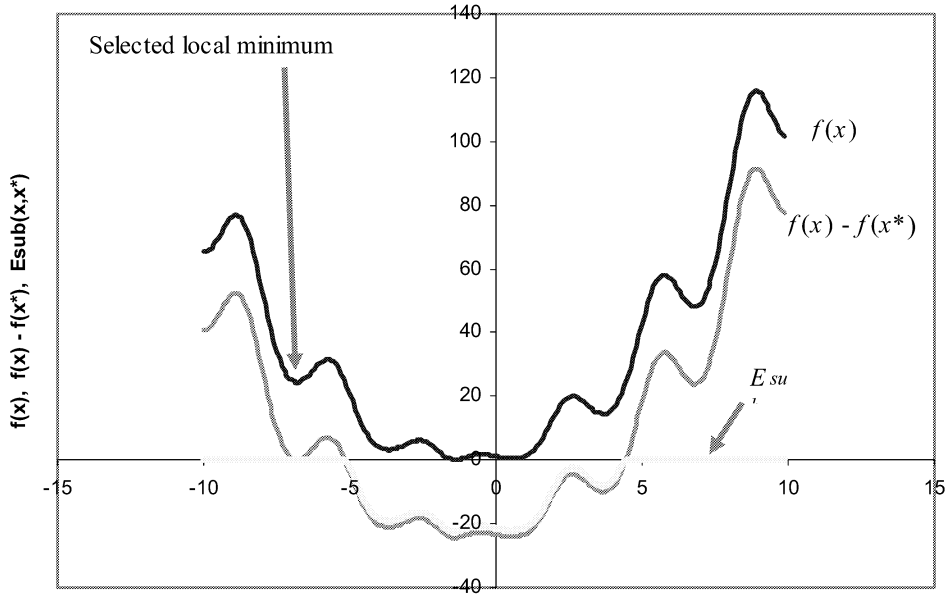
E_{sub} also has the same critical points as $f(\mathbf{x})$ and the same relative ordering of local minima, since

$$\frac{\partial E_{sub}(x,x^*)}{\partial x} = 0 \quad \Leftrightarrow \quad \frac{\partial f(x)}{\partial x} = 0 \tag{7}$$

and

$$\frac{\partial E_{sub}(x,x^*)}{\partial x} \cong \begin{cases} 0 & f(x) \geq f(x^*) \\ \partial f(x)/\partial x & f(x) < f(x^*) \end{cases} \tag{8}$$

Figure 1. Illustration of Subenergy Tunneling for x* = - 6.80678 f* = 24.411



TRUST solves Eq (2) using an indirect approach. More precisely, rather than minimizing directly $f(\mathbf{x})$, TRUST defines a new (*virtual*) objective function $E(\mathbf{x},\mathbf{x}^*)$.

Let $\hat{f}(\mathbf{x}) = f(\mathbf{x}) - f(\mathbf{x}^*)$, and H denote the Heaviside function. Then,

$$E(\mathbf{x},\mathbf{x}^*) = E_{sub}(\mathbf{x},\mathbf{x}^*) + E_{rep}(\mathbf{x},\mathbf{x}^*) \tag{3}$$

$$E_{sub} = \log_e\{1/[1+e^{(-\hat{f}(\mathbf{x},\mathbf{x}^*)-a)}]\} \tag{4}$$

$$E_{rep} = -\frac{z}{z+1}\rho H(\hat{f}(\mathbf{x},\mathbf{x}^*))\sum_{n=1}^{3N}(x_n - x_n^*)^{(z+1)/z} \tag{5}$$

The subenergy tunneling transformation E_{sub} is a nonlinear monotonic transformation that has several useful properties. These include a filter effect and the preservation of critical points.

This is illustrated in Fig 1, above, for the sample function $f(x) = (\sin[2x] - x - 1)^2$. For the selected local minimum *x**, the original function, its shifted value, and its subenergy tunneling transform are plotted. The filter effect and the preservation of critical points are clearly illustrated.

The virtual objective function (3) is a superposition of two contributing terms. Its effect is to transform the current local minimum of $f(\mathbf{x})$ into a global maximum, while preserving any lower lying local minima. This is illustrated, below, in Fig.2.

The actual search for $\mathbf{x}_g$ is carried out in terms of the flow of a system of coupled nonlinear ODEs constructed from the virtual objective function. Specifically, in order to achieve a reduction of *E* as the simulation evolves, one needs to enforce $dE/dt < 0$. Thus,

$$\dot{\mathbf{x}} = -\frac{\partial E(\mathbf{x}, \mathbf{x}^*)}{\partial \mathbf{x}} \qquad (9)$$

which results in the coupled differential equations

$$\dot{x}_n = -\frac{\partial f(\mathbf{x})}{\partial x_n} \frac{1}{1 + e^{f(\mathbf{x}) - f(\mathbf{x}^*) + a}} + \rho (x_n - x_n^*)^{1/z} H[f(\mathbf{x}) - f(\mathbf{x}^*)] \quad (10)$$

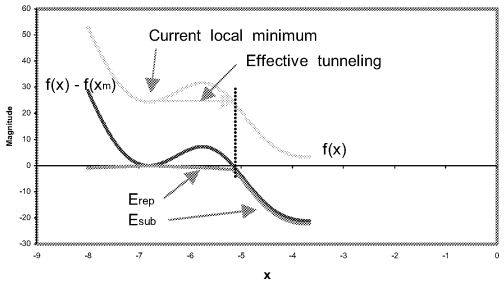
Each equilibrium state of this system will be a local minimizer of E, hence a local or global minimizer of f.

In Eq (10), the adaptive shift parameter *a* is recalculated whenever the tunneling solution emerges into a new (lower) basin of attraction. It enables the smooth transition between tunneling and descent to a lower local minimum. The quantity *z* appearing in the exponent in Eqs (5,10) is a positive odd integer called the Zak parameter [Barhen *et al* 1989]. As illustrated in Fig 2, gradient descent applied to $f(\mathbf{x})$ and initialized at $\mathbf{x}^*+\varepsilon$ can not escape from the basin of attraction of $\mathbf{x}^*$. However, whenever gradient descent is applied to $E(\mathbf{x}, \mathbf{x}^*)$ and initialized at $\mathbf{x}^*+\varepsilon$, it always escapes the basin. In that sense, TRUST has a global descent property.

3. The TRUST Simulation Code

The modeling and simulations reported herein were carried out using the TRUST code, which is written in Visual FORTRAN 95. This modern version of the FORTRAN language was chosen in preference to commercial packages (such as MATLAB, SIMULINK, MATHEMATICA, etc) because of the considerable gain in computational speed. Moreover, the TRUST visualization routines create detailed graphics displays that allow, in *real time*, to follow the evolution of a simulation. FORTRAN95 was also preferred to the C++ language for

Figure 2. Illustration of TRUST Paradigm



the same reasons FORTRAN77 remained the first choice of scientists and engineers over PL-I in the 1970s, namely execution speed, compactness, and transparency to the user.

Figure 3, below, corresponds to a direct screen dump at the conclusion of a simulation. It comprises three regions. The upper region displays the successively generated values of the function to be minimized along with the current lowest local minimum. The color-coded magnitudes are plotted versus the number of function evaluations.

The lower left region displays the magnitude of the corresponding largest gradient components. As the simulation progresses, one is able to monitor the convergence status. The lower right region shows the projection on a plane of the solution path in phase space.

The computational kernel at the heart of a TRUST simulation has the following structure. Following appropriate initialization procedures, an iterative search is performed based upon the solution of Eq (10). At each iteration, TRUST first evaluates $f(\mathbf{x})$ and its gradient.

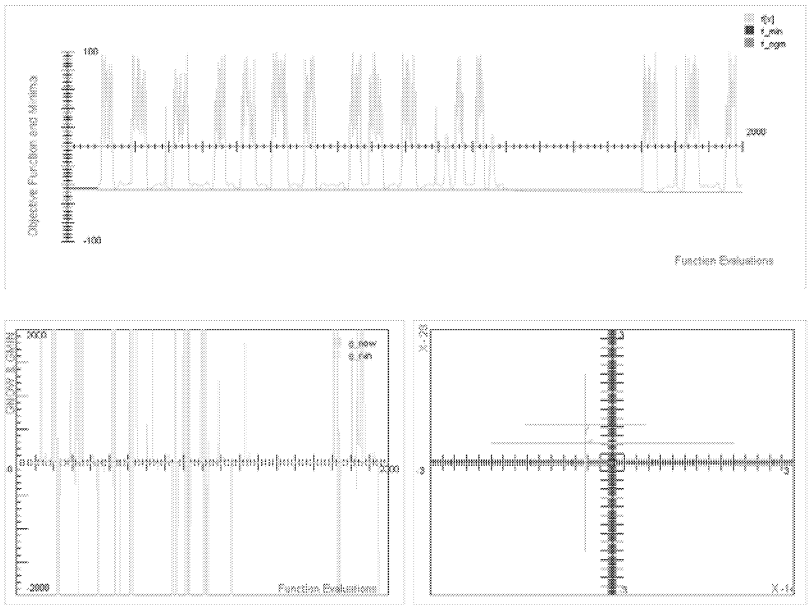


Figure 3. Direct screen dump from a TRUST simulation. On the upper half of the screen, the actual function values are in green. The tracking estimate of the global minimum is the lower (blue) curve.

The gradient is computed synergistically with $f(\mathbf{x})$. This can be done analytically (in this study), or using an automated differentiation procedure [Barhen and Reister 2003]. No finite differences are used. Hence, it entails only a small overhead cost.

For each successive point on the energy landscape hypersurface, TRUST then selects the search mode from five currently implemented alternatives. These include the detection of convergence to a local minimum, the detection of a plateau of $f(\mathbf{x})$, the detection of a well profile, descent into a new basin of attraction, or sub-energy-surface tunneling. When a plateau is detected, a bounded discrete ray tracing algorithm is executed. When a well profile is detected, descent is performed by a new, fast fractional power adaptive switching gradient descent algorithm. Specifically, one uses

$$x_n^{\mu+1} = x_n^{\mu} - \Delta_t w_n^{\mu} \left(\frac{\partial f(\mathbf{x}^{\mu})}{\partial x_n}\right)^{1/z} \qquad (11)$$

for gradient magnitudes less than one, where

$$w_n^{\mu} = 2^{-g(n,\mu)}. \qquad (12)$$

The function $g(n,\mu)$ is incremented whenever a gradient component changes sign. The updated information is interfaced to the graphics routines for real-time display. Then, the need for reflections from the domain boundaries is examined, and stopping tests are performed. Finally, key quantities of interest are updated and, if appropriate, the next iteration is initiated.

4. Results

Selected results of our simulations are shown in Tables 1 and 2, and in Figure 4. In Table 1, *N* denotes the number of atoms in the cluster and *NFGM* is number of function evaluations TRUST needs to reach the global minimum. For the reported results, we used the initial positions given by the Cambridge Table [Cambridge 2004].

N	Cambridge Table	TRUST	Difference	*NFGM*
8	-19.821489	-19.821651	0.000162	32
32	-139.635524	-139.63998	0.004456	123
38	-173.928427	-173.93428	0.005853	152
46	-220.680330	-220.68845	0.00812	68010
60	-305.875476	-305.88788	0.012404	176
75	-397.492331	-397.50919	0.016859	855
86	-465.384493	-465.40527	0.020777	62397
98	-543.665361	-543.69189	0.026529	115044
99	-550.666526	-550.69238	0.025854	394

Table 1. Comparison of minimal energies of various clusters as reported in the Cambridge table and obtained by TRUST.

The results obtained to date show that TRUST discovered global minima that are consistently lower than those determined by previously reported methods. The advances are rather small percentage-wise – which, judging by the existing literature, is quite the norm for this problem – but they are significant, since they identify TRUST as one of the leading engines for discovering conformational structures in computational chemistry and materials science.

Coordinates from Cambridge table (N=8) (x, y, z)			Coordinates found with TRUST (N=8) (x, y, z)		
0.243304	1.021828	-0.29574	0.24203	1.02189	-0.29396
-0.42129	-0.55447	0.682804	-0.42184	-0.55224	0.68835
-0.86119	-0.2132	-0.15128	-0.86197	-0.21272	-0.14630
0.553196	-0.32323	0.632042	0.55262	-0.32123	0.63672
0.715876	0.146823	-0.23513	0.71485	0.14725	-0.23126
-0.15897	0.214897	-0.71963	-0.15998	0.21393	-0.71584
0.056999	-0.59222	-0.18612	0.05617	-0.59192	-0.18064
-0.12792	0.299569	0.273058	-0.12860	0.30084	0.27670

Table 2. Coordinates of the atoms at the global minimum in a cluster with $N = 8$ atoms.

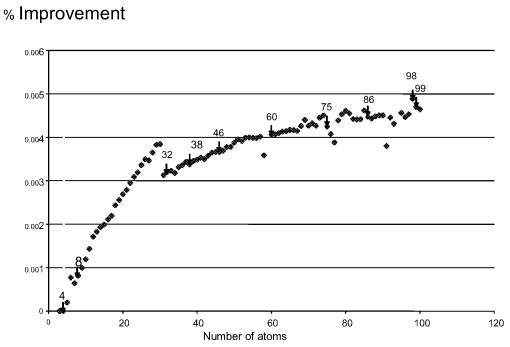


Figure 4. Improvement (percentage-wise) of results obtained with TRUST as a function of *N*. The arrows indicate the clusters reported in Table 1.

This significance is enhanced by the fact that the number of function evaluations required to obtain these minima is, in general, much lower than in other approaches. The improvement curve shown in Fig. 4 indicates clearly two different regimes, corresponding to small and intermediate clusters, respectively. While this feature is not surprising, we intend to investigate it more thoroughly and determine its potential relevance for the GO problem. Substantial effort will also be focused on assessing the robustness of TRUST's performance with respect to initial conditions.

ACKNOWLEDGEMENTS. This research was performed at the Center for Engineering Science Advanced Research in the Computer Science and Mathematics Division, Oak Ridge National Laboratory. Funding was provided by the Materials Science and Engineering Program of DOE's Office of Basic Energy Sciences. Additional support (for CR) was received from the Research Alliance for Mathematics and Science, DOE Office of Advanced Scientific Computing Research. ORNL is operated by UT-Battelle LLC, under contract DE-AC05-00OR22725.

References

Barhen, J., S. Gulati and M. Zak, "Neural learning of constrained nonlinear transforma-tions", *IEEE Computer*, **22**(6), 67-76 (1989).

Barhen, J., V. Protopopescu, and D. Reister, "TRUST: a deterministic algorithm for global optimization", *Science*, **276**, 1094-1097 (1997).

Barhen, J. and D. Reister, "Uncertainty analysis based on sensitivities generated using automated differentiation", *Lecture Notes in Computer Science*, **2668**, 70-77, Springer (2003).

Cai, W., H. Jiang, and X. Shao, "Global optimization of Lennard-Jones clusters by a parallel, fast annealing, evolutionary algorithm", *J Chem Inf Comp Sci*, **42**, 1099-1103 (2002).

Cambridge, http://www.wales.ch.cam.ac.uk/~jon/structures /LJ/tables.150.html

Deaven, D., N. Tit, J. Morris, and K. Ho, "Structural optimization of Lennard-Jones clusters by a genetic algorithm", *Chem Phys Lett*, **256**, 195-200 (1996).

Hoare, M. and P. Pal, "Statistics and stability of small cluster nuclei", *Nature Physical Science*, **230**, 5-8 (1971).

Hoare, M. and P. Pal, "Geometry and stability of spherical fcc microcrystallites", *Nature Physical Science*, **236**, 35-37 (1972).

Johnston, R., "Evolving better nanoparticles: genetic algorithms for optimizing cluster geometries", *Royal Society for Chemistry Dalton Trans*, 4193-4207 (2003).

Leary, R., "Global optima of Lennard-Jones clusters", *J Global Optimization*, **11**, 35-53 (1997).

Leary, R., "Global optimization of funneling landscapes", *J Global Optimization*, **18**, 367-383 (2000).

Lee, J., I.-H. Lee, and Jo. Lee, "Unbiased global optimiza-tion of Lennard-Jones clusters for $N \leq 201$ using the conformational space annealing method", *Phys Rev Lett*, **91**(8), 080201 1-4 (2003).

Locatelli, M, and F. Schoen, "Efficient algorithms for large-scale global optimization: Lennard-Jones clusters", *Comp Optim and Applic*, **26**, 173-190 (2003).

Northby, J., "Structure and binding of Lennard-Jones clusters: $13 \leq N \leq 147$", *J Chem Phys*, **87**(10), 6166-6177 (1987).

Reister, D., J. Barhen, J. DuBose and E. Oblow, "Global optimization to maximize stack energy", *Geophysics*, **66**(1), 320-326 (2001).

Wales, D. and J. Doye, "Global optimization by basin hopping and the lowest energy structures of Lennard-Jones clusters containing up to 110 atoms", *J Phys Chem*, **101 A**, 5111-5116 (1997).

Wales, D. and H. Scheraga, "Global optimization of clusters, crystals, and biomolecules", *Science*, **285**, 1368-1372 (1999).

Wolf, M. and U. Landman", "Genetic algorithms for structural cluster optimization", *J Chem Phys*, **102 A**, 6129-6137 (1998).

Xue, G., "Improvement on the Northby algorithm for molecular conformation: better solutions", *J Global Optim*, **4**, 425-440 (1994).

Author Short Bios

Dr. JACOB BARHEN is the Director of the Center for Engineering Science Advanced Research (CESAR) at the Oak Ridge National Laboratory (ORNL) and a UT-Battelle *Corporate Fellow*. From 1987 to 1994, Dr. Barhen was the head of the Neural Computation and Nonlinear Science Group at Caltech / JPL. He began his career at ORNL, where he headed the Machine Intelligence Group from 1978 to 1987. Dr. Barhen's research interests include (*i*) global optimization; (*ii*) neural networks; *(iii)* emerging high performance computer systems; (*iv*) sensitivity and uncer-tainty analysis of complex models, and (*v*) optical information processing. He has authored over 160 scientific papers, holds 8 U.S. Patents, has received 3 Space Act awards for major contributions to the US Space Program, 11 NASA awards for technical innovation, as well as an R&D100 award. Dr. Barhen received his D.Sc. degree from the Technion-Israel Institute of Technology, Haifa, in 1978. He is a member of the AAAS, IEEE, SPIE, the International Neural Networks Society, and the Planetary Society. E-mail: barhenj@ornl.gov .

Dr. VLADIMIR PROTOPOPESCU is a distinguished research staff at CERSAR/ORNL, where he has been working since 1985. Dr. Protopopescu's interests include: (i) control techniques for distributed systems; (ii) global optimization; (iii) time serial analysis for nonlinear systems, and (iv) modeling of complex systems. He has authored over 170 scientific papers and two books, holds two US Patents, and received the Physics Award of the Romanian Academy as well as an R&D 100 Award. Dr. Protopopescu received his Ph.D. from the Institute of Atomic Physics, Bucharest, Romania, in 1976. He is a member of the APS, AMS, SIAM, and IAMP. E-mail: vvp@ornl.gov .

JULIEN DAHAN was born in 1982 in Sens, France and went to Paris to study mathematics, physics and chemistry at the Ecole Supérieure de Physique et Chimie Industrielles (ESPCI), from which he will receive a M. Sc. Degree in 2006. He is currently an intern at CESAR/ORNL. E-mail: julien.dahan@bde.espci.fr .

GABRIEL AMSELEM was born in 1983 in Paris, France and has studied at the ESPCI for the past two years. He expects to graduate from ESPCI in 2006 with a M.Sc. Degree. He is currently an intern at CESAR/ORNL. E-mail: gabriel.amselem@bde.espci.fr .

CHRISTOPHER RANDALL will graduate in 2005 from Alabama A&M University with a B.Sc. in Mechanical Engineering. As part of the DOE/SC/RAMS program, he was a summer 2004 intern at CESAR/ORNL. His permanent e-mail is: crandall9123@yahoo.com .

ZACHARY VANE, a native of Oak Ridge, TN is an undergraduate student in Engineering at Virginia Polytechnic Institute. He was a summer 2004 intern at CESAR/ORNL His e-mail is: zvane@vt.edu .

A SIMULATION-BASED VALUE STREAM MAPPING FORMAL METHOD FOR LEAN IMPLEMENTATION: ATOMS, DATABASE AND MODEL GENERATOR

Yang-Hua Lian, Hendrik Van Landeghem
Department of Industrial Management
Ghent University
Technologiepark, 903, B-9052, Ghent, Belgium
email: yanghua.lian@ugent.be, hendrik.vanlandeghem@ugent.be

KEYWORDS
Lean, Value stream mapping, Object-oriented simulation, Model generator, Reuse, Database

ABSTRACT

Value Stream Mapping (VSM) has become a popular lean implementation method in recent years. However, its drawbacks such as being time-consuming and its inability to detail the dynamic behavior of production processes have stimulated some researchers to look for tools to enhance VSM. Simulation, which creates consensus by its model-building interface, visualization of dynamic views and output analyzing abilities has been proposed as such a tool. In this paper, we define a simulation modeling method for lean implementation called ″simulation-based VSM formal method″. Atoms, database structure and model generator are designed and created. An example applying this formal method is illustrated.

I. Introduction

Lean transformation, such as physical factory layout redesigns or changes of the supply chain infrastructure, involve high costs. In recent years, VSM has emerged as the preferred way to implement lean. VSM is a mapping tool used to describe supply chain networks. It maps not only material flows but also information flows that signal and control the material flows (Rother and Shook 1999). However, since VSM is a paper and pencil tool of lean manufacturing, it is time-consuming. Furthermore, VSM is a static tool which has limitations for detailing dynamic behaviors, disability to handle the complexity and uncertainty, and causes barriers to effectively apply the improved situations. We have tried to tackle these drawbacks and proposed a 4-phases simulation-based value stream mapping method for lean implementation in our previous research (Lian and Van Landeghem 2002). In this paper, we further define the framework and elements of this simulation-based VSM modeling method and illustrate it with an example. Via the combination of simulation tool and VSM, managers can see the impact of lean before the implementation and transform the organization into a lean one at minimal cost.

Owing to supply chain integration, the traditional tasks of manufacturing systems design are becoming more complex. Alfieri and Brandimarte (1997) indicate that the variety of tasks involved calls for a modular approach which is characterized by reuse modules and stable interfaces to connect them and enables the designer to experiment with different alternatives by simply assembling a set of predefined building blocks. Objected-oriented simulation languages are well-suited to the purposes of the approach. Furthermore, as indicated in the literature (Thomson, 1994; Drake et al., 1995; Son et al., 2003), simulation models have been termed ‘throw away models’ because they are seldom used after the initial plans or designs are finalized. Valentin and Verbraeck (2001, 2002) has also pointed out that the traditional simulation studies fail to support model reusability and maintainability, the reduction of the model size and complexity as well as modeling time reduction. An object-oriented modeling approach can attain all these advantages. Based on these reasons, we have chosen an objected-oriented simulation language, Taylor Enterprise Dynamics® (Taylor ED) as our modeling language.

In Taylor Enterprise Dynamics®, everything is an atom. Atoms are the object-oriented modeling resources. A product, a server, a queue, a model, a library and even the software itself are all atoms of Taylor ED. In addition to Taylor ED’s standard atom libraries, users can create new atoms themselves. A C-like programming language called 4DScript (4D refers to 4 dimensions: x, y, z and time) is used to write the coding which controls the behavior of atoms.

We have designed 11 atoms for VSM in our research till now. The idea is that all the VSM icon functions can be corresponding by composing the atoms we design and some ED standard atoms. Then, simulation models of different VSM scenarios can be generated automatically by composing those from a set of atoms with a designed database and a model generator. The model generator (named "model builder" in Son et al., 2003) is designed to create a discrete-event simulation model for a predetermined simulation language from the neutral description of the system and the actual data in the database so that the system is data-driven (Son et al., 2003). The framework of our formal method is shown in Figure 1.

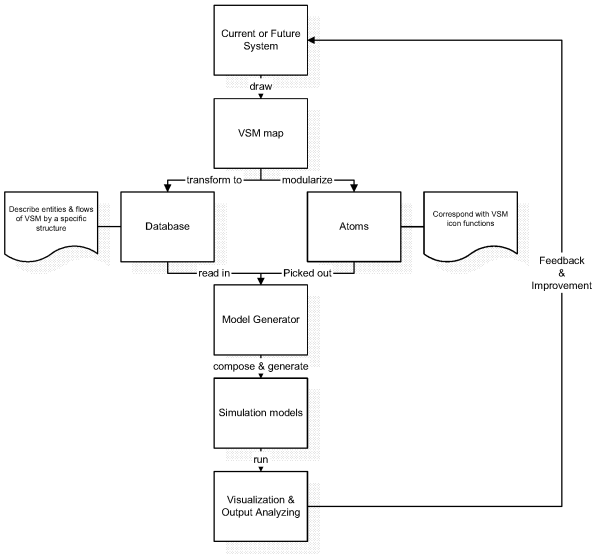


Figure 1 Formal method framework

II. Literature Review

Some researchers have proposed simulation as a tool to enhance value stream mapping for lean manufacturing (McDonald, Van Aken and Butler, 2000; Lian and Van Landeghem 2002). Simulation has been classified into five approaches by Son et al. (2003): 1) natural language interfaces, 2) graphical interfaces, 3) interactive dialogue interfaces, 4) resource and process descriptions, 5) the use of production-resource and execution models. All simulation packages use some combination of these approaches. The literature categorized by Son et al. is summarized in Table 1.

Although Oldfather (Oldfather et al., 1966) developed the earliest simulation generator in 1966, the concept of object-oriented modeling (component-based modeling) or applying both object-oriented modeling and generators to simulation rised in the middle of year 90's due to supply chain integration, increase of manufacturing system design's complexity and drawbacks of traditional simulation studies.

Alfieri and Brandimarte (1997) use the objected-oriented simulation language, MODSIM, as their performance evaluation environment and in order to design the building blocks properly, a modeling methodology at a higher level, OMT (object modeling technique) methodology, is proposed. The object-oriented modeling approach makes both the documentation and the transition from conceptual modeling to simulation code easier. A simple multi-echelon inventory management system case is considered with the aim of illustrating the use of inheritance (in a hierarchical class structure where subclasses inherit the features of their parents) in object modeling.

Aytug and Dogan (1998) introduce a generalized framework for the definition of kanban-controlled manufacturing systems and a simulation generator based on this framework. Five major building blocks (work center module, production kanban monitoring module, withdrawal kanban monitoring module, material order module and customer order module) of the simulation code that will be used to model a manufacturing system are developed as well as a database and a user interface are designed and implemented. A hypothetical manufacturing system is used to demonstrate the simulation generator. This generator is an enhanced version of Christenson and Dogan (1995). In Christenson and Dogan (1995), the system assumes that all parts must follow the same process routing. In Aytug and Dogan (1998), the generator can handle convergent and divergent flows, and different parts can have different routings. Few specific limitations like deterministic setup times and transportation times and no formal modeling concepts in the generator for material handling devices are indicated.

Son, Jones and Wysk (2003) describe a library of such formal, neutral models of simulation components for a discrete-event simulation of the flow of jobs through a job shop based on a pre-provided schedule. Seven components (header, experiment, shop floor, product/process, production, station/schedule and output) are described. Each component in the library functions as a basic building block (module) to compose models of manufacturing systems of interest. A database structure is derived from the formal models which were created in EXPRESS and EXPRESS-G (Schenk and Wilson, 1994). Two model builders (generators), which compose the components and generate simulation models in terms of the neutral description of the system and the actual data in the database for two specific commercial packages (Arena and ProModel) are designed, respectively. However, the methodology is limited to discrete part manufacturing systems that operate with a single unit load and limited to a push environment which would require major modification when changing to a pull or a hybrid push/pull environment. The lack of material handling equipment and material handling/storage system components would need to be improved.

Either the ideas of self-contained modules (Alfieri and Brandimarte, 1997; Valentin and Verbraeck, 2001; 2002) and simulation generators (Oldfather et al., 1966; Mathewson, 1984; Lee, 1996; Aytug and Dogan, 1998; Son et al., 2003) or the combination of both (Aytug and Dogan, 1998; Son et al., 2003) are not new in the simulation, but our idea of applying these two concepts to enhance value stream mapping for the purpose of lean implementation is an initial in the research. The formal method we introduce is a discrete-event simulation-based method, which does what VSM can do, compensates for the drawbacks of static VSM, and generates the model automatically without constraints such as single unit load, capacity limitations, non-stochastic times, specific to push environment... etc. that have been addressed in the literature. We will also include an additional aspect, the aggregation of the data for real world to a representative simulation model, which was not considered in the research up to now.

Table 1 Simulation approaches (Son et al., 2003)

Approach	Characteristics	Authors	Interact system	Simulation code
Natural language	A natural language interface plus the input data and specifications to generate the simulation code	Heidorn, 1974	NLPQ	GPSS
		Ford & Schroer, 1987	EMSS	SIMAN
		Yuan, Dogan & Viegelahn, 1993	FORTRAN	SIMAN
Graphical Interface & Dialogue monitor	Describe the layout of components within a system by a graphical interface. Often combine with a natural language processor or a dialogue monitor	Doshi, Madala & Sinclair, 1985	N.A.	N.A.
		Murray & Sheppard, 1988	N.A.	N.A.
Resource & process models	Build models based on the process flow model and resource models	Lee, 1996	N.A.	N.A.
		Christenson, Dogan, 1995	N.A.	N.A.
		Aytug, Dogan, 1998	C++	SIMAN
Production-resource & execution models	Generate simulation models automatically via a formal production resource model and a finite-state-automata-based execution model	Son, 2000	Arena	SIMAN
		Son, Wysk, Jones, 2003	Arena	SIMAN

III. Atom Formalism and Manipulation

We create atoms mainly in terms of VSM icons. Atoms in a simulation model can be interpreted as VSM icons in a value stream map. They are object-oriented, self-contained modules of a simulation model and describe the behavior of entities (e.g. machine, product, buffer, storage, etc.) within a manufacturing system. The combination of atoms by connecting the links between the channels of the atoms, namely, building the simulation models according to entity relationships, can provide the dynamic flow information. The self-contained, reusable, replaceable characteristics of atoms aid to conduct a very effective way of simulation. They are called "atoms" in Enterprise Dynamics® (Incontrol Enterprise Dynamics 2001), "templates" in Arena ® from Systems Modeling Corp (Kelton et al. 1998), "building blocks" by Valentin and Verbraeck (2001, 2002), "simulators" by Hamoen and Moens (2002), "neutral components", "modules" or "objects" by Son et al. (2003) and so forth.

We have created 11 atoms through our research till now: VSMSUPR02 (Supermarket), VSMPROC01 (Manufacturing Process), VSMSUPP01 (Supplier), VSMSTAG01 (Inventory), VSMCUST01 (Customer), VSMOPRO01 (Order Process/Withdraw), VSMKSTA01 (Kanban Staging), VSMKWAL01 (KanbanWall), VSMSTOR01 (Buffer or Safety Stock), VSMPALL01 (Pallets), VSMDATA01 (Data Measurements). Due to space limit of the paper, we only introduce the formalism and manipulation of supermarket atom VSMSUPR02.

1. Supermarket atom formalism

The supermarket atom is a buffer between two machines or processings. It is ready to ship the parts or products when receiving a signal from a demand trigger or downstream processing; empty kanbans h (h=[x_i/M], an integer) would then be released to upstream to replenish the supermarket. As shown in Figure 2, we can divide the flows into two parts. The first part, demand delivery, is the downstream of supermarket atom. When a demand (outside trigger) comes into the system, it requests the order quantity x_i from a supermarket atom through an order-process/withdraw atom. If there is enough kanbans (size=M and number=k, total number of parts or components in the supermarket=M×k) in the supermarket atom, the demand trigger will cause a demand delivery or a machine processing before delivery (remaining kanbans=k–x_i/M). Otherwise, the demand will be put in a backlog queue. The second part, kanban replenishment, is the upstream of the supermarket atom. The empty kanbans of the supermarket are put in a staging queue and go into withdraw atom one by one after the previous empty kanban is filled up. For each empty kanban, there is a production kanban released to the production kanban wall. The upstream manufacturing process atom (machine or work center) receives orders from the production kanban wall to process components and send out its end-products to replenish the taken parts of the empty kanban (withdrawal kanban) waiting in the withdraw atom. The replenished full kanban is then sent back to the supermarket. The manufacturing process atom has a variable process time which is characterized by certain distribution function. In addition, a set-up time is considered in the processing machine between the fabrications of two different types of products in the case of multiple supermarket trays. The relationships among supermarket atom with other atoms are displayed simply in Figure 3 using ER-diagram of Chen (Chen, 1976).

2. Supermarket atom manipulation

We follow the manner of Enterprise Dynamics atom reference guide v 5.0 (Incontrol Enterprise Dynamics, 2001) and guide the users for atom manipulation by main points like atoms description, keywords, input/output channels, right / double click and atom status, which are shown in Table 2.

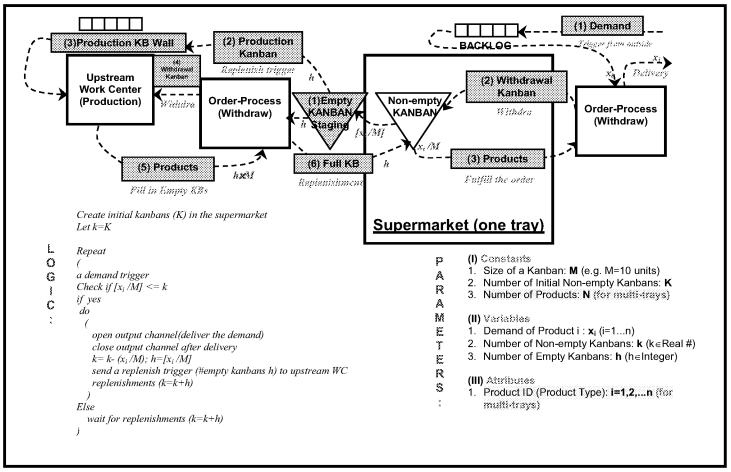


Figure 2 Supermarket (VSMSUPR02) atom formalism

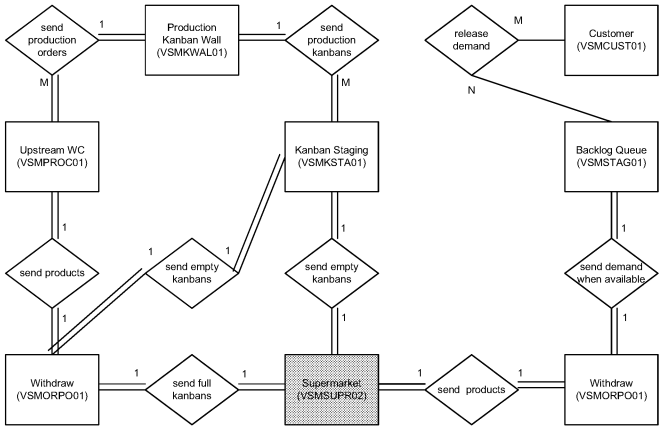


Figure 3 A simple ER-diagram of atoms related to supermarket atom using Chen (Chen, 1976) notations

Table 2 Supermarket atom manipulation reference guide

Atom Description

This atom is used to model supermarket of a pull system that helps signal demand for the product. In a supermarket, a fixed amount of raw material, work in process, or finished product is kept as a buffer to schedule variability or an incapable process. The products inside the kanban are released from second output channel of supermarket atom. When a kanban is empty, it is sent out from the first output channel. The replenished full kanbans are received through the only input channel.

Keywords

Inventory, buffer, pull system, order, replenish

Input Channels

1 minimum, 1 maximum

Input channel 1: Receive full kanbans. Connect to an atom that sends out full kanbans to replenish supermarket.

Output Channels

2 minimum, 2 maximum

Output channel 1: Release empty kanbans. Connect to a staging atom (VSMKSTA01) where these empty kanbans stay temporarily.

Output channel 2: Release products. Connect to an atom (order-process/withdraw atom, VSMORPO01) that "pulls" parts from kanban.

Right Click / Double Click

- *Atom Name:* A name given by the user to represent the entity
- *Icon:* 2D icon selection
- *Creation Trigger:* A field to add some functions to the kanbans being created in the supermarket
- *Number Kanbans (Initial/Constant):* A integer number given by the user to decide number of kanbans in the supermarket
- *Kanban Size:* A integer number given by the user to know how many products are contained by a kanban
- *ProdType:* A label to distinguish products in case of multi-tray supermarket

Status

empty/full/not empty

IV. Database Structure

In the simulation-based VSM formal method that we propose, a model generator written in Enterprise Dynamics programming language 4DScript reads the data from a database with several tables (a Microsoft Access database in our application), takes the required atoms, connects them and generates the simulation models through a wholly automatical process. It reads the entity types/names, layout/location of the entities, the relationships between the entities, and transforms all these messages into a simulation model. This database describes the resources/entities, their layout and relationships within a manufacturing system and contains all the information which is mentioned in a VSM. A database structure is designed and displayed in Figure 4. Different data information provided under the same structure can create various simulation models with a unique model generator. When executing the generated simulation model, output measurements of products or orders such as lead time, value-added time, value-added ratio... are written to two output tables within the same database. The database tables can be classified into four categories and we will illustrate few of database tables in the section of hypothetical example.

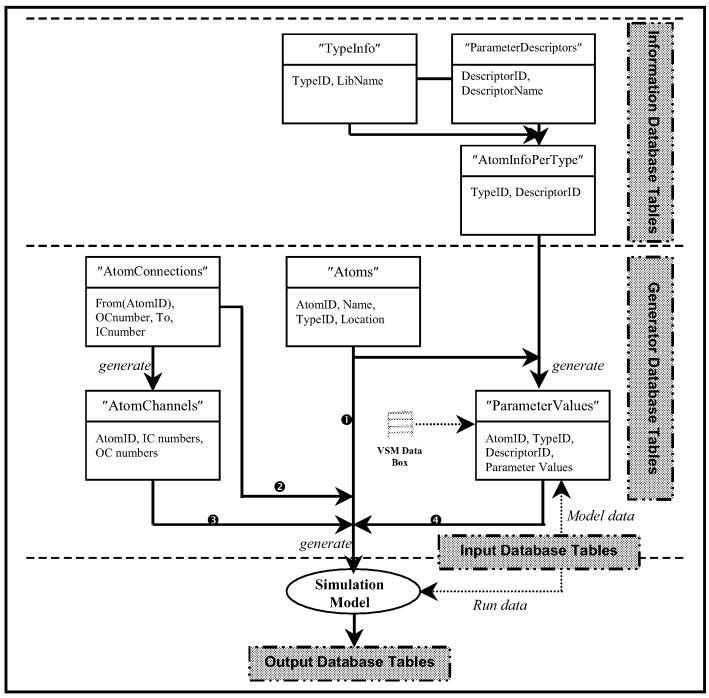


Figure 4 Database structure

1. Generator database tables

This type of database tables describes the resources (entities or atoms), their layout location (X, Y, Z) and relationships (connections) within a manufacturing system. Via the information in the "Connections" table, a "Channels" table is generated also by the generator to provide the quantity information of input and output related entities for each specific entity. The generator needs all these tables to select required atoms, put them at the right location, open channels, connect related atoms and fill in atom parameters. We can say these tables are functional tables which depict the entities and contain the logic of a simulation model. *The data within these tables change from models to models.* There are four tables belonging to this category: "Atoms", "AtomConnections", "AtomChannels" and "ParameterValues" tables.

There are some rules to be followed to fill the data for the four generator database tables (Figure 4):

- "Atoms" table (❶): data can be obtained from a VSM map according to icons and their layout by (1) decide VSM icons and their corresponding atoms, (2) give atomID for each atom, (3) give names for each atom (workstation), (4) decide location of each atom according to physical factory layout.
- "AtomConnections" table (❷): data can be gained according to the VSM flow.
- "AtomChannels" table (❸): data are generated from table ❷. If table ❷ is available, the generation of this channels table is just an action of click.
- "ParameterValues" table (❹): parameter values can be filled according to the data provided by VSM data box.

For the users who are not familiar with ED Taylor, it would be rather difficult to fulfill the mission of filling data for these database tables from a VSM map, especially for the "AtomConnections" table (❷) that the users are requested to have certain experience of ED to convert VSM flows into atom connections. In the further step of our research, we will develop algorithms and program them to let these data tables being automatically generated from the messages contained in a value stream map.

2. Information database tables

There are three information tables ("TypeInfo", "ParameterDescriptors", "AtomInfoPerType"). They give the data information about a list of atoms being using in the simulation model, a list of parameters for all those atoms and a list of parameters for each type of atom. The tables are used to provide further information for the values filled in the generator tables. The generator does not query them and *the data within the tables do not change from models to models.* Thus, they are three fixed tables and the users have no need to fill the data for them.

3. Input database tables

These tables provide data values for the specific table possessed by an atom. For instance, the atom "VSMOPRO01" has a bill of material table; the atom "T098-Table" which is an ED standard atom can be used as order data or setup time matrix table. The data values required of these atom tables are read in from this sort of input database table and the data of input database table should be provided by the users themselves (e.g. order data, BOM, setup times).

4. Output database tables

Two tables are filled with values when a measurement atom "VSMDATA01" is used in the model. It gives output data of product arrival/departure time, product lead time, value-added time and ratio in one table and order arrival/departure time, order lead time, due time (in seconds) in another table. When order lead time is larger than the due time, it means the delivery is not on time. We can evaluate our system by output measurements of these output tables plus the standard output data supported by ED itself such as machine utilization, queue size, queuing time, etc. The function of calculating work-in-process (WIP) of the whole system will be added to the features of the atom soon.

V. Model Generator

The model generator reads the data from a structured database, takes the required atoms (entity types/names), locates the position (layout/location of the entities), connects them (the relationships between the entities) and transforms all the messages from the database into a simulation model automatically. The model generator is data-driven so that different data source under the same database structure will produce various simulation models with the only generator.

Our model generator is written in 4Dscript in a *.fnc file which is a function file of ED. In order to execute the model generator function in *.fnc file easily, we also change the *.mnu file of ED which describes all the options that have been added to the menu of the application. We add the menu "Model / Import Model / ***(function name of the model)" and the simulation model can be generated by selecting the menu or by shortcut key such as "Shift+Ctrl+B" if also being assigned in *.mnu file. A small part of the model generator program is shown in Figure 5.

Simulation models are the final products of atoms, database and model generator. By changing the data in the database, we can yield the simulation model which corresponds to the new data set by just one shot (e.g. point to the menu option or push the shortcut keys). In this way, different scenarios of VSM can be transformed into simulation models in short time and we can easily obtain feedbacks and improvements to the various VSM scenarios after analyzing and comparing the outputs of simulation models.

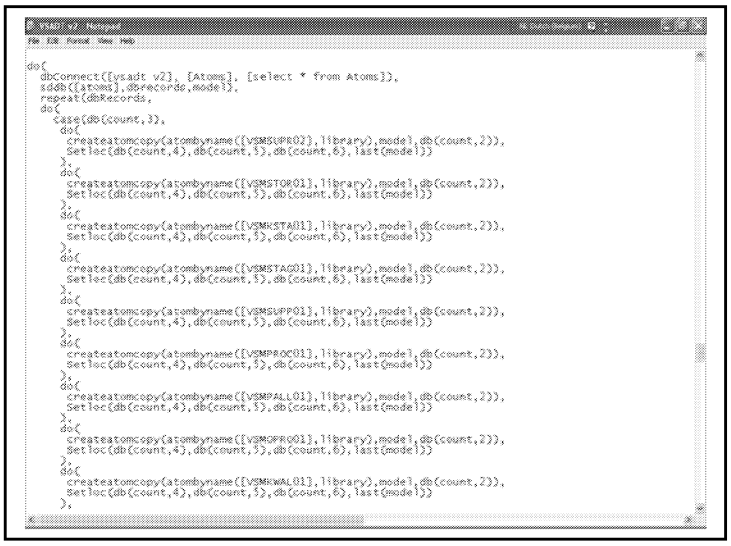


Figure 5 A small part of model generator program

VI. A Hypothetical Example

A hypothetical VSM map like Figure 6 is used to illustrate our approach. There are two products produced in the factory. Three kinds of materials are ordered from outside suppliers: material of product 1, material of product 2 and the common material used for both product 1 and 2. Only material of product 1 and product 2 need to be cut. After cutting, the materials are deposited in kanban size of 37.5 kg in the supermarket (1 kanban for each product type). Common material is used in the stage of mixing which mixes material of product 1 or 2 with common material in certain percentage. For example, every kilogram of final product 1 is made of 600 g of material product 1 (60%) and 400 g of common material (40%). After mixing, the work-in-process products are weighed and stored in kanban size of 180 kg in the supermarket (4 kanbans for each product type).

When an order arrives, the three filling and packing work center (big fill machine, small fill machine and manual filling) start the jobs according to the final product size that the customer wants. If the customer ordered 200 kg product 1 in big size, the big filling and packing machine will withdraw 200 kg from the supermarket, fill and pack in big size and deliver to the customer. At this moment, there are one empty kanban and three non-empty kanbans (160kg, 180kg, 180kg) in the supermarket. One production kanban will be released to upstream workstation (mix and weigh) for replenishment of the empty kanban. The mix workstation will withdraw the materials from its upstream supermarket and the common material storage. The empty kanbans of this upstream supermarket will be replenished by releasing production kanbans to its upstream cutting workstation and so on. Each workstation has a process time and setup times. * symbol in Figure 6 means that the setup times should be referred to setup time matrix tables.

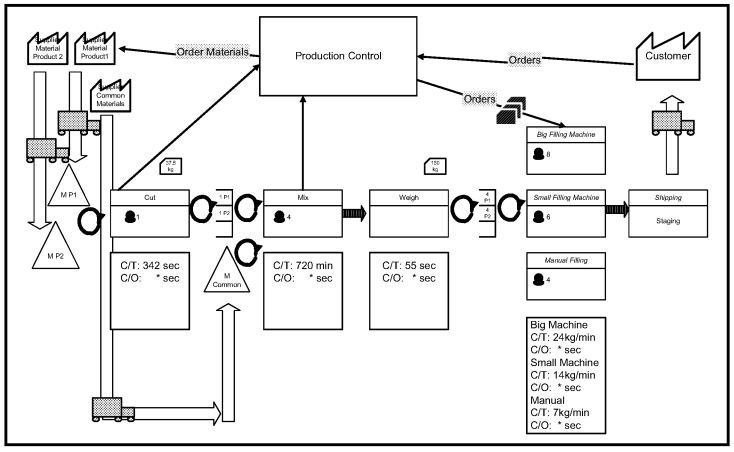
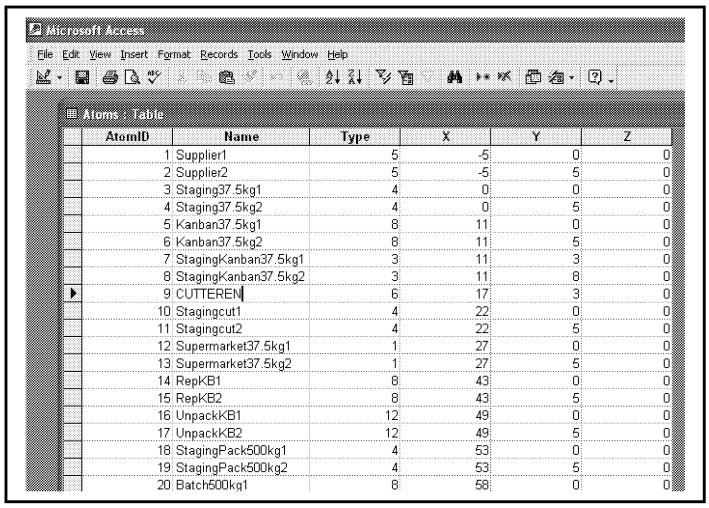


Figure 6 Hypothetical example VSM map

Four generator database tables, "Atoms", "AtomConnections", "AtomChannels" and "ParameterValues", are illustrated according to the information provided by the hypothetical example.

- "Atoms" table: It gives the information about the atoms needed for the simulation model. For instance, a cutting machine in the factory can be represented as a manufacturing process atom called VSMPROC01 in the library with TypeID 6 (atom type IDs are listed in the information database table "TypeInfo" which is not displayed due to space limit of the paper). We give it AtomID 9 and it is named "CUTTEREN" in the model. Its location (X, Y, Z) is (17,3,0). (See Figure 7, record 9).
- "AtomConnections" table: This table describes the relationships with other atoms for each atom. In Figure 8, we can see that the output stream of AtomID 12 which is a supermarket atom has relation with other two atoms (ID 7, 14). Its two output channels should be connected to input channel 1 of AtomID 7 which is a kanban staging atom (VSMKSTA01), and to input channel 2 of AtomID 14 which is order-process/withdraw atom (VSMORPO01) such as we have mentioned in Table 2 of atom reference.

- "AtomChannels" table: This table is generated from "AtomConnections" table. It tells the quantity of related atoms in terms of input and output stream for each specific atom. For example, AtomID 1 has only one atom related to it of input stream and so does the output stream. AtomID 12, a supermarket atom, has 1 atom related to its input stream and 2 atoms related to its output stream (Figure 9).
- "ParameterValues" table: This table gives the parameter values. In Figure 10, records 61-65 describe the parameter values for AtomID 12 which belongs to TypeID 1. In the information database table " AtomInfoPerType" (which is not shown in the paper), atom TypeID 1 has 5 parameters indexed as 1 (Initial Kanbans), 2 (Kanban Size), 3 (Constant Kanbans), 4 (Product Type), 17 (Creation Trigger). As we can read from Figure 10 of "ParameterValues" table, their values are 1 (1 initial kanban), 75 (kanban size 75 units), 1 (1 constant kanban), 1 (Product Type 1), 0 (no action for creation of initial kanbans).

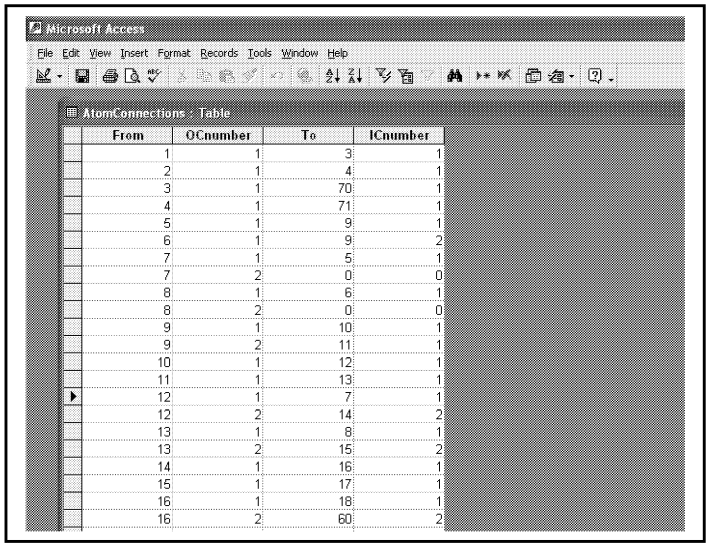

Microsoft Access

File Edit View Insert Format Records Tools Window Help

Atoms : Table

AtomID	Name	Type	X	Y	Z
1	Supplier1	5	-5	0	0
2	Supplier2	5	-5	5	0
3	Staging37.5kg1	4	0	0	0
4	Staging37.5kg2	4	0	5	0
5	Kanban37.5kg1	8	11	0	0
6	Kanban37.5kg2	8	11	5	0
7	StagingKanban37.5kg1	3	11	3	0
8	StagingKanban37.5kg2	3	11	8	0
9	CUTTEREN	6	17	3	0
10	Stagingcut1	4	22	0	0
11	Stagingcut2	4	22	5	0
12	Supermarket37.5kg1	1	27	0	0
13	Supermarket37.5kg2	1	27	5	0
14	RepKB1	8	43	0	0
15	RepKB2	8	43	5	0
16	UnpackKB1	12	49	0	0
17	UnpackKB2	12	49	5	0
18	StagingPack500kg1	4	53	0	0
19	StagingPack500kg2	4	53	5	0
20	Batch500kg1	8	58	0	0

Figure 7 "Atoms" database table

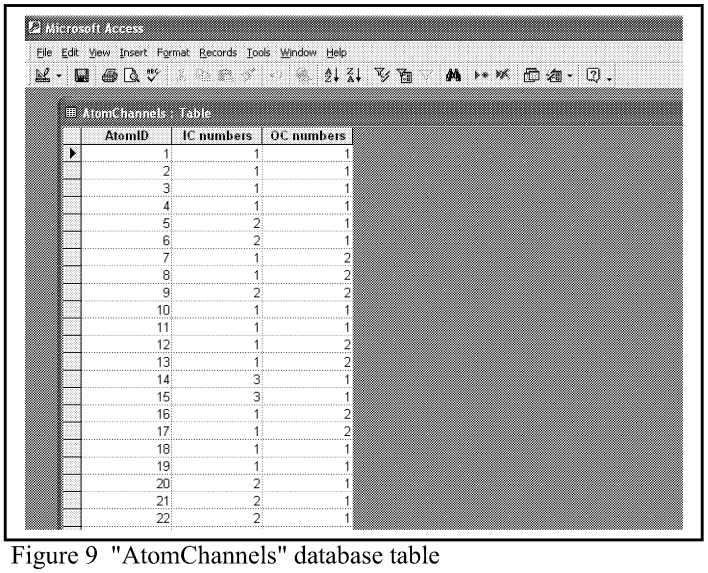

Microsoft Access

File Edit View Insert Format Records Tools Window Help

AtomConnections : Table

From	OCnumber	To	ICnumber
1	1	3	1
2	1	4	1
3	1	70	1
4	1	71	1
5	1	9	1
6	1	9	2
7	1	5	1
7	2	0	0
8	1	6	1
8	2	0	0
9	1	10	1
9	2	11	1
10	1	12	1
11	1	13	1
12	1	7	1
12	2	14	2
13	1	8	1
13	2	15	2
14	1	16	1
15	1	17	1
16	1	18	1
16	2	60	2

Figure 8 "AtomConnections" database table

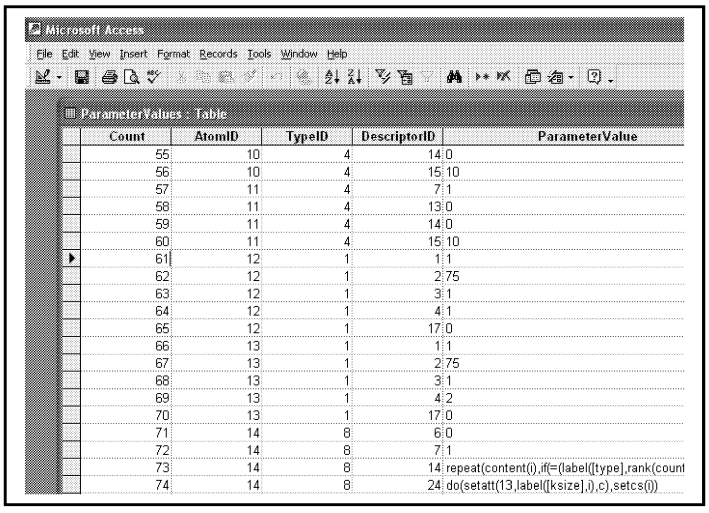

Microsoft Access

File Edit View Insert Format Records Tools Window Help

AtomChannels : Table

AtomID	IC numbers	OC numbers
1	1	1
2	1	1
3	1	1
4	1	1
5	2	1
6	2	1
7	1	2
8	1	2
9	2	2
10	1	1
11	1	1
12	1	2
13	1	2
14	3	1
15	3	1
16	1	2
17	1	2
18	1	1
19	1	1
20	2	1
21	2	1
22	2	1

Figure 9 "AtomChannels" database table


Microsoft Access

File Edit View Insert Format Records Tools Window Help

ParameterValues : Table

Count	AtomID	TypeID	DescriptorID	ParameterValue
55	10	4	14	0
56	10	4	15	10
57	11	4	7	1
58	11	4	13	0
59	11	4	14	0
60	11	4	15	10
61	12	1	1	1
62	12	1	2	75
63	12	1	3	1
64	12	1	4	1
65	12	1	17	0
66	13	1	1	1
67	13	1	2	75
68	13	1	3	1
69	13	1	4	2
70	13	1	17	0
71	14	8	6	0
72	14	8	7	1
73	14	8	14	repeat(content(i),if(=(label([type],rank(count
74	14	8	24	do(setatt(13,label([ksize],i),c),setcs(i))

Figure 10 "ParameterValues" database table

After executing the model generator, a simulation model is generated automatically in terms of the data set from the database tables. Figure 11 is a partial scene of the running generated simulation model which focuses on the pull flow of supermarket with kanban size 180kg and the histograms of output measurements. One of the output measurements tables is shown in Figure 12 and we can see that order-17 and order-19 are not delivered on time (order lead time > due time).

VII. Conclusions and Further Research

Simulation tools have been proposed and used to enhance value stream mapping recently. In this paper, we introduce a simulation modeling method for lean implementation called simulation-based VSM formal method. Atoms formalism and manipulation, database structure and model generator are explained and the formal method is applied to a hypothetical example. To our knowledge, the approach of applying "building blocks" and "simulation generator" concepts to strengthen value stream mapping for the purpose

of lean implementation constitutes new research results. The formal method is a discrete-event simulation-based method, has no limitations of single unit load, non-stochastic times and can be used to model either push or pull manufacturing system.

Further research will focus on: 1) create a few more atoms of VSM icons (e.g. MRP atom of production scheduling, WIP atom to measure the work-in-process of the entire manufacturing system), 2) develop algorithms of converting the data of a VSM map into ″generator database tables″ and the transform procedure can be done automatically by just giving a command, 3) conduct some case studies by applying the method.

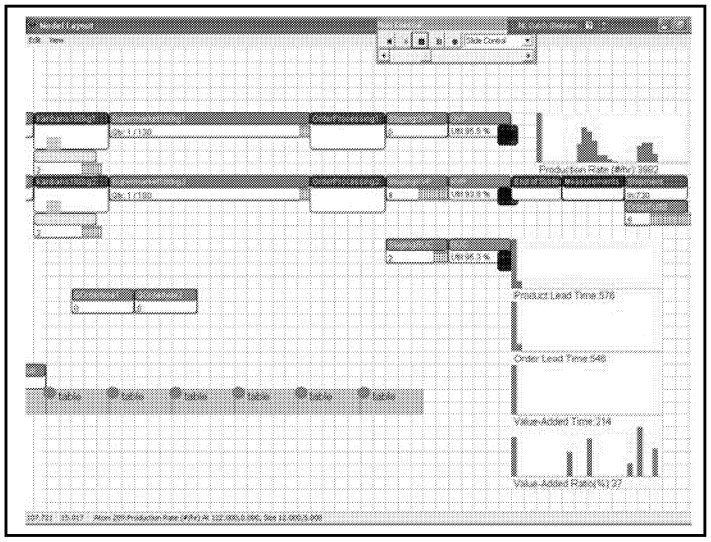
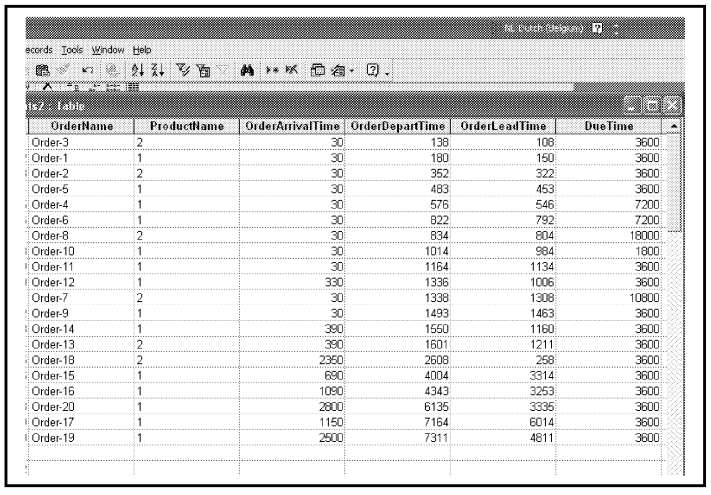


Figure 11 A partial scene of the running generated simulation model



ecords Tools Window Help

tv2 : Table

OrderName	ProductName	OrderArrivalTime	OrderDepartTime	OrderLeadTime	DueTime
Order-3	2	30	138	108	3600
Order-1	1	30	180	150	3600
Order-2	2	30	352	322	3600
Order-5	1	30	483	453	3600
Order-4	1	30	576	546	7200
Order-6	1	30	822	792	7200
Order-8	2	30	834	804	18000
Order-10	1	30	1014	984	1800
Order-11	1	30	1164	1134	3600
Order-12	1	330	1336	1006	3600
Order-7	2	30	1338	1308	10800
Order-9	1	30	1493	1463	3600
Order-14	1	390	1550	1160	3600
Order-13	2	390	1601	1211	3600
Order-18	2	2350	2608	258	3600
Order-15	1	690	4004	3314	3600
Order-16	1	1090	4343	3253	3600
Order-20	1	2800	6135	3335	3600
Order-17	1	1150	7164	6014	3600
Order-19	1	2500	7311	4811	3600

Figure 12 Output database table – "OutputMeasurements2"

REFERENCES

Alfieri, A. and Brandimarte, P., 1997. "Object-oriented modeling and simulation of integrated production/distribution systems". *Computer Integrated Manufacturing Systems*, Vol. 10, No. 4, 261-266.

Aytug, H. and Dogan, C., 1998. "A framework and a simulation generator for Kanban-controlled manufacturing systems". *Computer and Industrial Engineering*, 34(2), 337-350.

Chen, P., 1976. "The entity relationship model – towards a unified view of data". *ACM Transaction on database systems*, 1 (1), 9-16.

Christenson, K. and Dogan, C., 1995. "A simulation generator for dual-card Kanban-controlled flow shops". *International Journal of Production Research*, 33(9), 2615-2631.

Doshi, K.A., Madala, S. and Sinclair, J.B., 1985. "GIST: A tool for specifying extended queueing network models". Technical Report TR8511, Rice University, Department of Electrical and Computer Engineering, Houston, Texas.

Drake, G.R., Smith, J.S. and Peters, B.A., 1995. "Simulation as a planning and scheduling tool for flexible manufacturing systems". *Proceedings of the 1995 Winter Simulation Conference*, IEEE, Piscataway, NJ, 805-812.

Ford, D. and Schroer, B., 1987. "An expert manufacturing simulation system". *Simulation*, 48(5), 193-200.

Hamoen, S.C. and D.J. Moens, 2002. "Logistic Simulator for Steel Producing Factories", *Proceedings of the 2002 Winter Simulation Conference*.

Heidorn, G.E., 1974. "English as a very high level language for simulation programming". *SIGPLAN Notices*, 9, 91-100.

Incontrol Enterprise Dynamics, B.V., Maarssen, The Netherlands. 2001. Enterprise Dynamics Software, *Enterprise Dynamics user manual*, 4th edition.

Kelton, D., Sadowski, D.A. and Sadowski, R.P., 1998. *Simulation with ARENA*, McGraw-Hill.

Lee, S., 1996. *Automatic Generation of Simulation Model for Shop Floor Control System*. Master's thesis, POSTECH.

Lian, Y.H. and Van Landeghem, H., 2002. "An Application of Simulation and Value Stream Mapping in Lean Manufacturing". *Proceedings of 2002 European Simulation Symposium*, Dresden, Germany.

Mathewson, S.C., 1984. "The application of program generator software and its extensions to discrete-event simulation modeling". *IIE Transactions*, 16, 3-18.

McDonald, T., Van Aken, E. and Butler, R., 2000. "Integration of Simulation and Value Stream Mapping in Transformation to Lean Production". *Proceedings of 2000 IIE Conference*.

Murray, K. and Sheppard, S., 1988. "Knowledge-based simulation model specification". *Simulation*, 50(3), 112-119.

Oldfather, P.M., Ginsberg, A.S. and Markowitz, H.M., 1966. "Programming by questionnaire: how to construct a program generator". Rand report, RM-5129-PR, IBM Corporation, LA, CA.

Rother, M. and J. Shook. 1999. *Learning to See: Value Stream Mapping to Add Value and Eliminate Muda*, Lean Enterprise Institute, Brookline, MA.

Schenck, D. and Wilson, P., 1994. *Information modeling the EXPRESS way*. New York, Oxford University Press.

Son, Y., 2000. *Simulation based shop floor control: Automatic model generation and control interface*. PhD dissertation, Penn State Univ.

Son, Y. J., Jones, A. T. and Wysk, R. A., 2003. "Component based simulation modeling from neutral component libraries". *International Journal of Computer and Industrial Engineering*, 45, 141-165.

Son, Y. J., Wysk, R. A. and Jones, A. T., 2003. "Simulation based shop floor control: formal model, model generation and control interface". *IIE Transactions*, 35(1), 29-48.

Thomson, M.B., 1994. "Expanding simulation beyond planning and design". *Industrial Engineering*, 26(10), 65-67.

Valentin, E.C. and Verbraeck, A., 2001. "The Use of the Building Blocks to Enhance Flexibility and Maintainability of Simulation Models and Simulation Libraries", *Proceedings of 2001 European Simulation Symposium*, Ghent, Belgium.

Valentin, E.C. and Verbraeck, A., 2002. "Guidelines for Designing Simulation Building Blocks", *Proceedings of the 2002 Winter Simulation Conference*.

Yuan, Y., Dogan, C. And Viegelahn, G., 1993. "A flexible simulation model generator". *Computer and Industrial Engineering*, 24(2), 165-175.

PARTITIONING AND MODULAR CODE SYNTHESIS FOR RECONFIGURABLE MECHATRONIC SOFTWARE COMPONENTS*

Sven Burmester† and Holger Giese
Software Engineering Group
University of Paderborn
Warburger Str. 100, D-33098 Paderborn, Germany
e-mail: [burmi|hg]@uni-paderborn.de

Alfonso Gambuzza and Oliver Oberschelp
Mechatronics Laboratory Paderborn
University of Paderborn
Pohlweg 98, D-33098 Paderborn, Germany
e-mail: [Alfonso.Gambuzza|Oliver.Oberschelp]@mlap.de

KEYWORDS
Hybrid Systems, Reconfiguration, Implementation.

ABSTRACT

Online reconfiguration offers great potential for improving the performance of technical systems at run-time. To enable reconfiguration, which can also be employed in complex hierarchical designs for mechatronic systems, support for the modeling and realization of modular reconfiguration is required. In this paper, we present the modular code synthesis for our modeling approach with Hybrid Components and a related Hybrid Statechart extension for the Unified Modelling Language (UML) which offers a representation of reconfigurable systems including conventional, non-reconfigurable block diagrams. To avoid the logic to be executed down to every possible ramification, the control behavior is at first partitioned into hierarchically organized continuous and discrete blocks. An implementation is then presented which schedules the evaluation of the continuous and discrete blocks in such a manner that the reconfiguration steps are safely and efficiently realized and the resulting code is suited to hard real-time systems.

INTRODUCTION

In mechatronic systems it is often necessary to adapt controllers to altered environmental conditions at runtime. Beside parameter adaptation it is in some cases necessary to alter the inner structure of the controller, for instance because of sensor failures. These structural changes have to be taken into account during modeling. If the structural variance is limited to predefined topologies in the first instance, the system is referred to as a reconfigurable one.

The inherent technical structure of the system can be used to define the reconfiguration. According to this approach a block-oriented model of a system consists of components describing a behavior (basic block) and of others describing a topology (hierarchy block). A hierarchy block describes relations and couplings between subordinated components while a basic block represents a directed input-output behavior in the shape of mathematical expressions which are specified by Ordinary Differential Equations (ODEs). The existing solvers for ODEs allow an application in hard real-time systems.

A reconfiguration is assumed to be an alteration of a hierarchy block. If possible configurations of a hierarchy are predefined, this is referred to as a statical reconfiguration. If possible configurations are assumed to be states of a hierarchy they can be represented as a statechart. The combination of statechart and configurable hierarchy is called a Hybrid Component.

As the reconfiguration is derived from the structure of the technical system, the principle of aggregation can be applied also to the information processing. In this case, each block makes up a module and the code can be generated separately. A hierarchy coordinates the exchange of data between modules and superior hierarchies.

A block diagram typically represents a forest of directed evaluation graphs by means of equations and couplings. If code is generated on the basis of this representation, the evaluation order of the equations of the system has to be calculated first. In the case of reconfigurable systems the evaluation order is usually not static, so a definite evaluation order is valid only for matching configurations. In combination with the module concept, code must be generated whose evaluation order can be controlled by the superior hierarchy.

The overall system makes up a so-called hybrid system. In the following, we will present the modular code synthesis for hybrid systems. In addition we will apply the modeling approach with Hybrid Components and a related Hybrid Statechart extension for the Unified Modelling Language (UML) which offers a representation of reconfigurable systems that includes classical, non-reconfigurable block diagrams [Giese et al., 2004, Burmester et al., 2004].

The paper is structured as follows: In the first section we introduce the testbed of a magnetic-levitation train and present the modeling of the controller switching. The

*This work was developed in the course of the Special Research Initiative 614 - Self-optimizing Concepts and Structures in Mechanical Engineering - University of Paderborn, and was published on its behalf and funded by the Deutsche Forschungsgemeinschaft.

†Supported by the International Graduate School of Dynamic Intelligent Systems. University of Paderborn

following section will present our partitioning approach which is essential for modular code synthesis. We will then detail the implementation techniques and give an overview of the related work. A conclusion and an outlook to future work will round off the paper.

EXAMPLE AND MODELING

As an example we take the control system of a testbed of a magnetic-levitation train (Figure 1). The testbed allows only a control of the body mass in vertical direction. It consists essentially of a supporting frame, two vertical guides for the body and carriage mass, sensors to obtain the position of the body and the carriage mass, two voice-coil actuators for the simulation of disturbances as well as the levitation-motor and the body and carriage mass connected by a mechanical spring. On the basis of this testbed we will show the modeling and code synthesis of a switchable control.

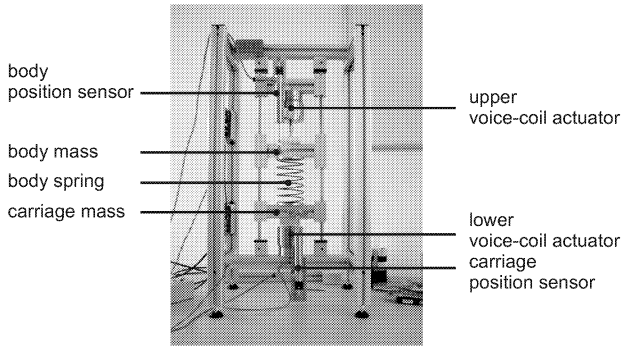


Figure 1: Testbed

Continuous Models

Multiple feedback controllers can be applied to the testbed. Figure 2 displays the three controllers that are applied. In Figure 2a the *Comfort* controller, providing the passengers the most comfort, is shown. It consists of two PIDT_1 controllers – one for controlling the undercarriage, the other for controlling the coach body. For inputs, $\text{PIDT}_{1,body}$ obtains the desired and the actual positions of the coach body. The first one is provided by a user input, the latter by a sensor. The output yields the position of the undercarriage and serves as an input for $\text{PIDT}_{1,carriage}$. The other input, the current position of the undercarriage, is provided by a sensor as well.

If, however, a user input does not exist, this value is set to a constant value as displayed in Figure 2b. In case the $x^{body}_{current}$ sensor should fail, this controller structure could lead to an instability. Then the system needs to be reconfigured as shown in Figure 2c. The required position of the undercarriage is set to a constant value. This

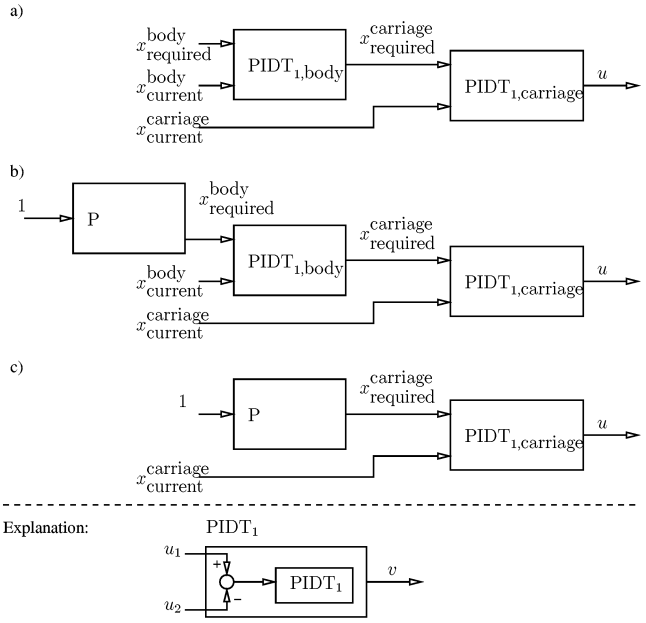


Figure 2: a) Comfort Controller, b) Semi-Comfort Controller, and c) Robust Controller

controller provides less comfort, but guarantees stability. To ensure stability, fault tolerance patterns are applied in such a way that the signal $x^{carriage}_{current}$ is never lost.

Hybrid Models

In order to model the exchange of single controller structures (reconfiguration) we use Hybrid Components and Hybrid Statecharts as described and formalized in [Giese et al., 2004, Burmester et al., 2004].

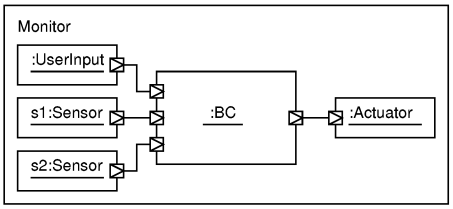


Figure 3: Structure of the controlling system

Figure 3 displays the structure of the controlling system. It consists of one Monitor component which embeds sensors, an actuator and a Body Control (BC) component; the latter contains the feedback controllers and switches between them.

The behavior of this hybrid BC component is specified by a simple Hybrid Statechart (see Figure 4) which is an extension of a Real-Time Statechart [Burmester and Giese, 2003]. It consists of 3 different control modes (discrete states) associated with the three configurations from Figure 2. The configurations consist of $\text{PIDT}_{1,carriage}$, $\text{PIDT}_{1,body}$, and the P blocks. Bold

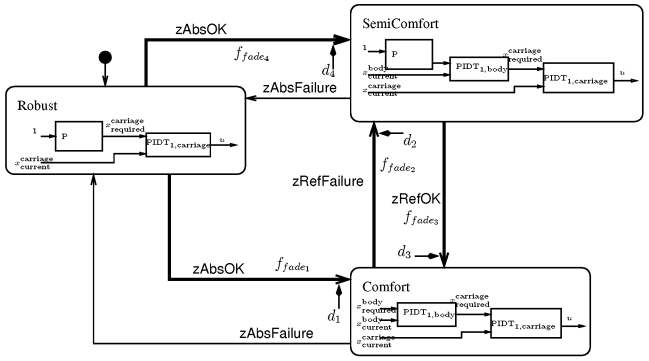


Figure 4: Behavior of the Body Control component

arrows indicate that output cross-fading has to be applied when there is a switch between two states. The deadline intervals d_i specify the minimum and maximum fading time allowed. Thin arrows indicate that a switch is performed without time-consuming fading.

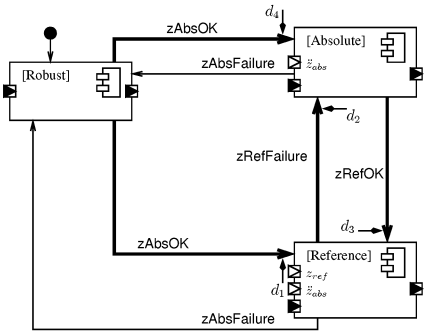


Figure 5: Interface Statechart of the Body Control component

When using this component in advanced contexts (e.g., embedding the component in another configuration), usually an abstract view of the component without the implementation details is sufficient. This view is given by the hybrid *Interface Statechart* of the component (see Figure 5). It consists of the externally relevant real-time information (discrete states, their continuous in- and outputs, possible state changes, their durations, signals to initiate transitions, and signal flow information [Oberschelp et al., 2004]). They abstract from the embedded components and from the fading-functions. Ports that are required in each of the three interfaces are filled in black, the ones that are only used in a subset of the states are filled in white.

In this example each discrete state of the component has a different continuous interface, a fact that leads to an Interface Statechart which consists of as many discrete states as does the detailed Hybrid Statechart. Usually not every internal reconfiguration will result in a different external state, which leads to further reduction of complexity in the Interface Statechart. The external view of the monitor component (see Figure 6) does not change at run-time although it consists of four discrete states. Therefore its Interface Statechart consists of just one state.

As displayed in Figure 3, the BC component is structurally embedded into the Monitor component. Figure 6 shows the behavior of the monitor embedding and coordinating the behavior of BC and the other embedded components as described in [Giese et al., 2004, Burmester et al., 2004]. For this embedding the notation from Interface Statechart (see Figure 5) is used.

The Hybrid Statechart in Figure 6 consists of four states representing that (i) a user input exists as well as the sensor providing $x^{body}_{current}$ (state AllAvailable), (ii) both of these signals are not available (state NoneAvailable) and (iii + iv) exactly one of these signals is available (states BodyAvailable and UIAvailable). Note again that the sensor providing $x^{carriage}_{current}$ is laid out as fault-tolerant.

Every state is associated with a configuration. Thus a switch from e.g. NoneAvailable to BodyAvailable results in a reconfiguration and a switch of the embedded BC component from state Robust to SemiComfort. The states UIAvailable and NoneAvailable are required to keep track of the available signals, yet a switch between them will not lead to a reconfiguration or further state switches.

Transitions, visualized by bold arrows, are associated with deadlines because they are time-consuming. They are triggered by events, raised from the xBody Sensor or UserInput component.

PARTITIONING

When the Monitor component is evaluated, it will trigger the evaluation of its embedded components. As not every embedded component belongs to every configuration, it depends on the current discrete state of the Monitor which of the embedded components are evaluated. Then the triggered components will themselves trigger their embedded components (in dependency of their discrete states) and so forth.

Thus there is one *evaluation order* per discrete global state. Enhancing the top-level monitor component with this information is usually not feasible as the number of global states grows exponentially with the number of components. Therefore we compose the whole system as a tree structure consisting of single Hybrid Components.

Each Hybrid Component is partitioned into multiple discrete and continuous evaluation nodes [Oberschelp et al., 2004]. The continuous evaluation nodes compute continuous states and outputs of the feedback controllers, the discrete evaluation nodes switch the discrete states of the components and reconfigure subordinated Hybrid Components. As the application of certain continuous evaluation nodes depends on the actual configuration and thus on the actual discrete state, safety can only be guaranteed if one continuous evaluation cycle is not preempted

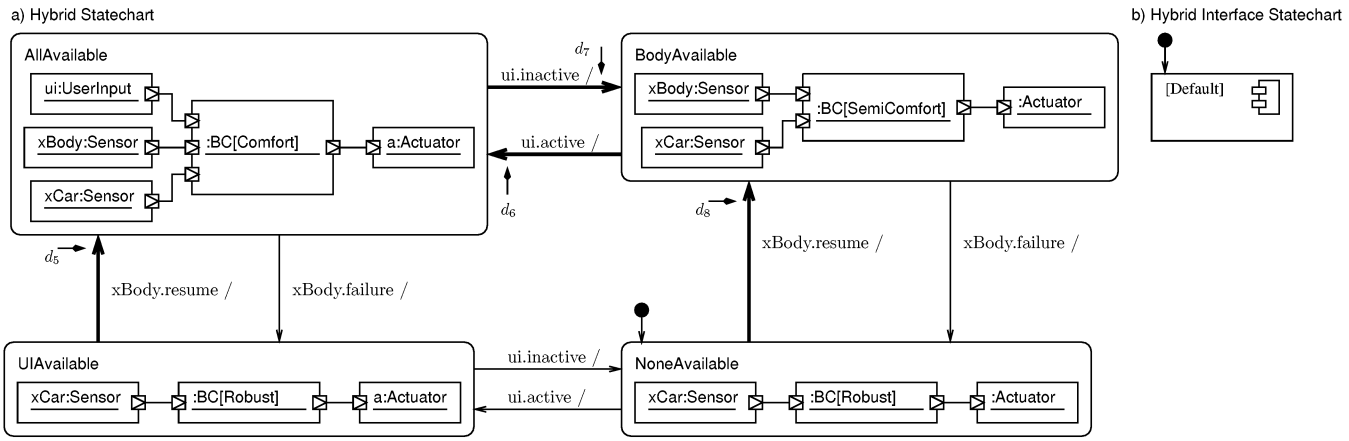


Figure 6: a) Behavior and b) Interface Statechart of the Monitor component

by the evaluation of a discrete node which switches the actual discrete state. Therefore we separate the evaluation of the discrete nodes from the evaluation of the continuous nodes in time.

Basic Continuous Components

As already said, we have to determine the *evaluation order* of the continuous nodes in dependency of their external couplings. In order to minimize computational effort, we partition multiple single evaluation nodes of subordinated components into evaluation nodes of the superordinated component. The order within such a superordinated evaluation node is static and thus does not change. We apply the partitioning algorithm from Appendix A which extends the partitioning algorithm from [Oberschelp et al., 2004] in order to build the superordinated evaluation nodes.

One of the nodes contains only expressions which influence only the inner continuous state of the controller. In [Oberschelp et al., 2004] this evaluation node is known as *state node (S)*. There is no particular evaluation order required for these expressions. Furthermore one evaluation node is created that determines outputs which do not depend on the current inputs (*non-direct link node (ND)*). Only the expressions whose outputs depend on the inputs (*direct link nodes (D)*) are partitioned into multiple evaluation nodes. The output of the algorithm is called the *reduced evaluation graph* of the configuration. It shows the interface of the component, the partitioned evaluation nodes, and their input-output dependencies. Figure 7a shows for example the P and a PIDT_1 controller. Each node represents a set of expressions (e.g., mathematical expressions) and the arrows indicate the dependencies. The reduced evaluation graphs, which are obtained from the application of the partitioning algorithm, are shown in Figure 7b. The P controller consists of one (direct link) node n_{d0}, the PIDT_1 controller consists of one (direct link) node n_{d0}

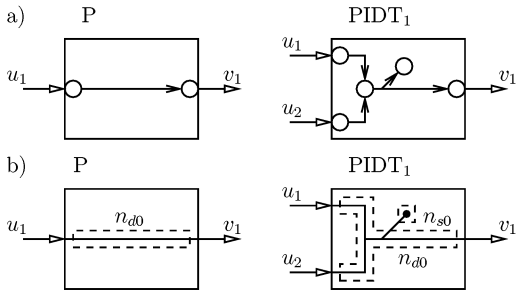


Figure 7: a) Evaluation node structure and b) *Reduced evaluation graph* of a P and a PIDT_1 controller

and a (state) node n_{s0}.

Basic Discrete Components

Discrete evaluation nodes do not need to be partitioned, because every discrete component consists of exactly one discrete node. In dependency of the current discrete state, transitions are checked for activation and – as the case may be – are fired in this evaluation node.

Hybrid Hierarchical Components

When the controller is embedded into a well-known configuration (that defines the external coupling of the single controllers), the algorithm is applied again to partition the configuration. Figure 8a shows the configuration of the SemiComfort state from Figure 4 (cf. Figure 2b). The partitioning results in the reduced evaluation graph of Figure 8b. Partitioning of the other configurations of Figure 4 (cf. Figures 2a and 2c) is done similar. Thus each of the three discrete states of the BC component is associated with another reduced evaluation graph.

As shown in Figure 6, the configuration that is associated with a discrete state does not have to consist exclu-

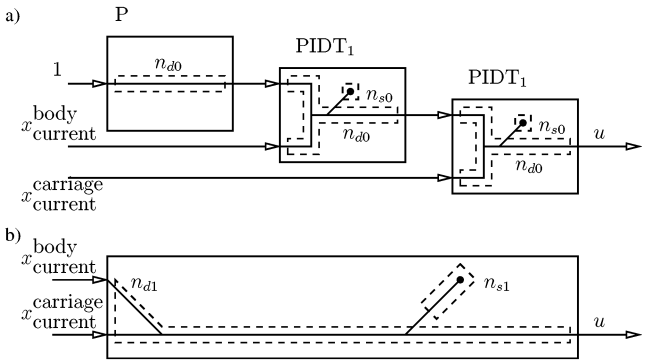


Figure 8: Configuration and reduced evaluation graph of the semi comfort controller

sively of continuous components, but may also consist of Hybrid Components. These Hybrid Components are in a specific discrete state in this configuration. Then the partitioning algorithm works on the reduced evaluation graphs of the appropriate discrete states.

When evaluating the discrete evaluation nodes of hierarchical components one has to make sure that a change at the top-level component affects the subordinated components within the same execution cycle. Therefore the discrete nodes of the components at a higher hierarchy level must be evaluated prior to the ones at the lower hierarchy levels.

IMPLEMENTATION

We have already stated that the evaluation order of the different nodes of a component depends on the external couplings. Thus an implementation must allow evaluation of the nodes in a different order, dependant on the couplings of the current configuration.

Basic Continuous Components

In order to evaluate the nodes in a different order each basic continuous component is implemented as a class, providing an evalCont(int nodeId) method. According to the parameter of the method, the corresponding node is evaluated (cf. Figure 9). The possible nodeIds n_{d0} or n_{d0}, and n_{s0} resp. correlate to the nodes which have been determined by the partitioning algorithm (cf. Figure 7b), the internals (auxiliars[0]=...) are the expressions of the components (cf. Figure 7a).

Basic Discrete Components

Similar to the implementation of a basic continuous component, a discrete component consists of a method evalDiscrete(). It has no nodeId parameter as it consists of exact one evaluation node. Furthermore it consists of an

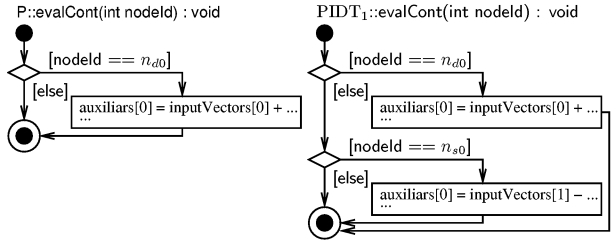


Figure 9: Activity Diagram of the evalCont(int nodeId) method of a P and a $PIDT_1$ controller

attribute current indicating the current discrete state.[1] Inside the evalDiscrete() method, there is a check – in dependency of the current discrete state – if transitions are triggered. In case of some triggered transitions, one is selected, its side-effects are executed and the discrete state is changed. This application flow is displayed in Figure 10.

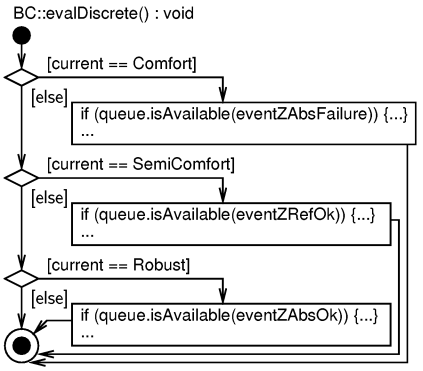


Figure 10: Activity Diagram of the evalDiscrete() method of BC

Hybrid Hierarchical Components

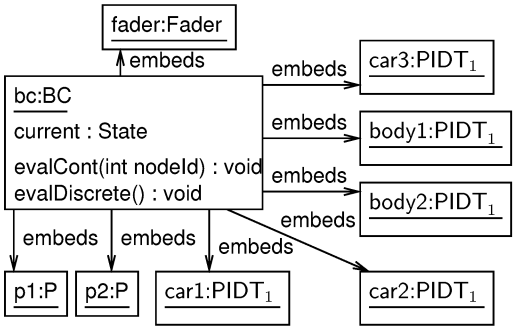


Figure 11: Object structure at run-time (cut-out)

An implementation of the hybrid hierarchical compo-

[1] In case of a flat automata model, current is usually of a simple data type, like int. In a statechart model with hierarchical and orthogonal states it is of a more complex type.

nents, like BC, consists of the methods evalCont(int nodeId) and evalDiscrete(). These methods call the according methods of the embedded components. Therefore the hierarchical component needs to have references to these components. Figure 11 displays this structure by a UML object diagram for the BC component.
The implementation of the evalCont(int nodeId) method of a hybrid hierarchical component differs from that of a basic continuous component. It contains not expressions but lists of continuous nodes of the *embedded* (basic continuous or hybrid hierarchical) components. These lists are dependent on the current discrete state. Figure 12 shows the content of the evalCont(int nodeId) method of BC as an activity diagram. In dependency of the current discrete state each node (n_{d1} and n_{s1}) consists of a different list of evaluation nodes of embedded components. If, for example, the component is in state SemiComfort, the node n_{d1} will consist of three sequential calls of the nodes n_{d0} from P controller p1, block n_{d0} from PIDT_1 controller body2, and block n_{d0} from PIDT_1 controller car2.
Note that the evaluation nodes of the embedded components, which are executed when n_{d1} or n_{s1} is evaluated, are determined by the partitioning algorithm as shown in Figure 8. These nodes of the subordinated components are the nodes from their corresponding reduced evaluation graphs. Thus no knowledge about the internal node structure of the embedded components is required inside BC.
Furthermore there exists source code for the so-called *fading states* FadeRobustComfort, etc. These states belong to the underlying hybrid automata model, as presented in [Giese et al., 2004]. The component resides inside these states while fading transitions are executed. Note that in this case the nodes n_{d1} and n_{s1} are similar to a concatenation of the according nodes of states Robust and Comfort and an additional evaluation node of a Fader component. Due to limited space, not all fading states are displayed in Figure 12.
The discrete evaluation node of BC is the node presented in Figure 10. As BC embeds only basic continuous components, no other discrete nodes have to be evaluated.
Figure 13 shows the implementation of the evalCont(int nodeId) method of the Monitor component in form of an activity diagram. Due to a lack of space we do not show that a Monitor instance references the embedded sensor and actuator components and the BC component (similar to Figure 11). As Monitor is a *self-contained* component it consists of exactly one continuous evaluation node n_{d2}. This node is periodically evaluated at run-time.

RELATED WORK

Due to an ever-increasing complexity of technical systems methods for modular simulation have been developed in recent years. To be emphasized in particular are descriptor methods based on differential-algebraic equations (DAE). One drawback of these methods is that they cannot be used in real time contexts because of the iterative parts of the necessary solvers. There are several CAE tools that are also based on DAEs, such as ADAMS[2] or SIMPACK[3] which allow multi-body modeling and simulation yet have the same drawbacks in real-time evaluation as do all DAE-based tools [Hahn, 1999]. MATLAB/Simulink and Stateflow[4] are the de facto industry standard for the modeling of technical systems. Reconfiguration can be modeled by *conditionally executed subsystems* which are executed in dependency of a control signal. Although this approach allows modeling of reconfiguration, it has the disadvantage that systems will become very complex. The models consist of all active and inactive components. The use of our notion of Hybrid Statecharts results in multiple configurations, consisting of just the active components.
Hybrid bond graphs [Mosterman and Biswas, 1995] introduce so-called *controlled junctions* to model reconfiguration. A finite-state machine (FSM) is associated with each controlled junction. Each state of the FSM is of the type on or off, indicating if the controlled junction acts like a normal junction or as a 0 value source. Therefore state changes turn parts of the model on or off. Modeling reconfiguration with hybrid bond graphs has the same drawback like conditionally executed subsystems, as graphs consist of all active and inactive configurations.
In [Mosterman, 2000] a Java and a C/C++ export to support simulation of hybrid bond graphs is outlined. The evaluation order has to be derived for every global state (the cross product of FSMs of all controlled junctions). In our approach, we exploit the hierarchical component structure to build a tree structure, that avoids getting a number of evaluation orders that is exponential in the number of states. Further, a switch of a discrete state in a FSM can trigger transitions in other FSMs. Therefore no upper bound is given, describing the number of transitions, which fire before the system reaches a consistent, stable state and continuous evaluation can proceed. In our approach the discrete and the continuous evaluation are decoupled and the upper bound of firing transitions is set by the hierarchy.

CONCLUSION AND FUTURE WORK

In this paper we presented a way to generate modular source code for reconfigurable, safety-critical systems. The reconfiguration was specified by our notion of Hybrid Components and Hybrid Statecharts.
The main problem to be solved in the implementation of feedback controllers and hybrid systems is the partitioning of the model into nodes and the determination of the

[2] www.adams.com
[3] www.simpack.com
[4] www.mathworks.com

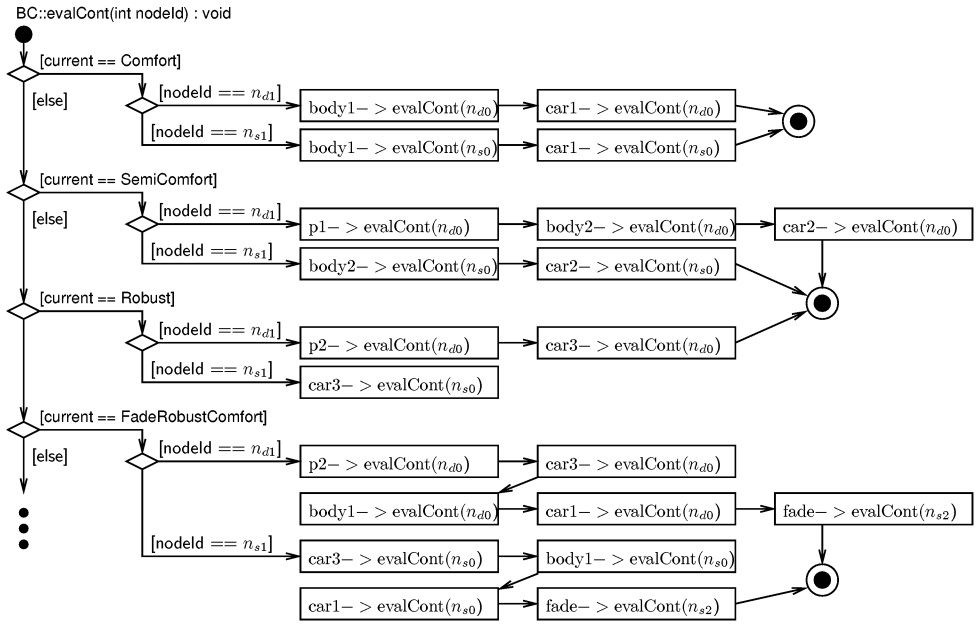


Figure 12: Activity Diagram of the evalCont(int nodeId) method of BC

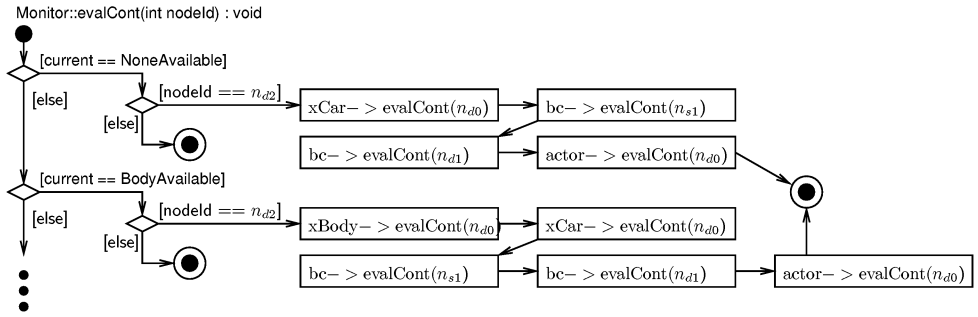


Figure 13: Activity Diagram of the evalCont(int nodeId) method of Monitor

node's evaluation order. In this paper we presented an advanced version of an algorithm that partitions basic continuous and discrete components, as well as hybrid hierarchical components. Then we showed how the partitioned model is mapped to modular structured code and presented a way of executing the code so as to ensure safe reconfiguration.

In current work we are implementing the automatic code generator and methods for the propagation of signals. Then we will test the presented example and evaluate the results.

ACKNOWLEDGEMENTS

The authors thank the students Vadim Boiko, Björn Schwerdtfeger, and Andreas Seibel for the interesting discussions.

REFERENCES

[Burmester and Giese, 2003] Burmester, S. and Giese, H. (2003). The Fujaba Real-Time Statechart PlugIn. In *Proc. of the Fujaba Days 2003, Kassel, Germany.*

[Burmester et al., 2004] Burmester, S., Giese, H., and Oberschelp, O. (2004). Hybrid UML Components for the Design of Complex Self-optimizing Mechatronic Systems. In *Proc. of 1st International Conference on Informatics in Control, Automation and Robotics (ICINCO 2004), Setubal, Portugal.* IEEE.

[Giese et al., 2004] Giese, H., Burmester, S., Schäfer, W., and Oberschelp, O. (2004). Modular Design and Verification of Component-Based Mechatronic Systems with Online-Reconfiguration. In *Proc. of 12th ACM SIGSOFT Foundations of Software Engineering 2004 (FSE 2004), Newport Beach, USA.* ACM. (accepted).

[Hahn, 1999] Hahn, M. (1999). *OMD - Ein Objektmodell für den Mechatronikentwurf. Anwendung in der objektorientierten Modellbildung mechatronischer Systeme unter Verwendung von Mehrkörpersystemformalismen. Fortschritt-Berichte VDI. Reihe 20. Nr. 299.* VDI Verlag, Düsseldorf.

[Mosterman, 2000] Mosterman, P. J. (2000). HYBRSIM - A Modeling and Simulation Environment for Hybrid Bond Graphs.

[Mosterman and Biswas, 1995] Mosterman, P. J. and Biswas, G. (1995). Modeling Discontinuous Behavior with Hybrid Bond Graphs. In *Proc. of the Intl. Conference on Qualitative Reasoning, Amsterdam, the Netherlands*, pages 139–147.

[Oberschelp et al., 2004] Oberschelp, O., Gambuzza, A., Burmester, S., and Giese, H. (2004). Modular Generation and Simulation of Mechatronic Systems. In *Proc. of the 8th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI), Orlando, USA.*

APPENDIX A: PARTITIONING ALGORITHM

A single node is characterized by input, output, and state variables and a set of expressions with a left-hand-side variable and a right-hand-side expression with references to other variables. We derive an acyclic expression graph $G = (N, E)$ with node set N and edge set $E \subseteq N \times N$, where for each $n \in N$ and the related expression $v := \ldots v' \ldots$ holds that for each variable v' the expression refers to in the right-hand-side an edge $(n', n) \in E$ exists with n' the node related to variable v'.

For an acyclic graph $G = (N, E)$ we have the following additionally defined terms: $d_{out}((N, E), n) := |\{n' \in N | (n, n') \in E\}|$ the out-degree of node n, $d_{in}((N, E), n) := |\{n' \in N | (n', n) \in E\}|$ the in-degree of node n, $N_{out} \subseteq N$ the subset of output nodes with $\forall n \in N_{out}$ holds $d_{out}((N, e), n) = 0$, $N_{in} \subseteq N$ the subset of input nodes with $\forall n \in N_{in}$ holds $d_{in}((N, E), n) = 0$, and $N_{state} \subseteq N$ the subset of nodes which represent the internal state of the block.

The partitioning problem for a given acyclic expression graph $G = (N, E)$ of a block is how to determine a minimum number of partitions $N_1, \ldots, N_n \subseteq N$ such that: $N = N_1 \cup \ldots \cup N_n$ and $\forall i \neq j \; N_i \cap N_j = \emptyset$, the derived graph $G_p = (N_p, E_p)$ with $N_p = \{N_1, \ldots, N_n\}$ and $E_p = \{(N_i, N_j) | i \neq j \wedge i, j = 1, \ldots, n \wedge \exists n' \in N_i \wedge n'' \in N_j : (n', n'') \in E\}$ is acyclic, and for any context graph $G' = (N', E')$ with $(N \cap N') \subseteq (N_{in} \cup N_{out})$ and $G'' = (N'', E'')$ with $N'' = N \cup N'$ and $E'' = E \cup E'$ an acyclic graph holds that the related derived graph for the partitioning build by $N_1, \ldots, N_n$ and each node of $N' - N$ is also an acyclic graph (cf. [Oberschelp et al., 2004]). Such a minimal partitioning can be computed by means of the following algorithm:

```
block_partitioning((N,E)) begin
  D_in : N → ℘(N_in ∪ N_state); // input dependencies
  I_out : N → ℘(N_out ∪ N_state); // influenced outputs
  L : N → ℘(N_in); // related input nodes
  c : N → ℕ; // visited successors

  // forward traversal to compute all input dependencies
  F := N_in ∪ N_state;
  forall n ∈ F do D_in[n] := { n }; done
  forall n ∈ N do c[n] := 0; done

  while ( F ≠ ∅ ) do
    forall (n ∈ F ) do
      forall n' ∈ N with (n,n') ∈ E do
        c[n'] := c[n'] + 1;
        if c[n'] == d_in((N, E), n') then
          D_in[n'] := ⋃_{(n'',n')∈ E} D_in[n''];
          if (n' ∉ N_state ∪ N_out) then
            F := F ∪ {n'};
          fi
        fi
      done
      F := F − {n};
    done
  done

  // backward traversal to compute all influenced outputs
  F := N_out ∪ N_state;
  forall n ∈ F do I_out[n] := { n }; done
  forall n ∈ N do c[n] := 0; done

  while ( F ≠ ∅ ) do
    forall n ∈ F do
      forall n' ∈ N with (n',n) ∈ E do
        c[n'] := c[n'] + 1;
        if c[n'] == d_out((N, E), n') then
          I_out[n'] := ⋃_{(n',n'')∈ E} I_out[n''];
          if (n' ∉ N_state ∪ N_in) then
            F := F ∪ {n'};
          fi
        fi
      done
      F := F − {n};
    done
  done

  // compute S and ND blocks
  N' := N; //nodes of the D blocks
  forall n ∈ N do
    // n is an element of the S-block
    if (I_out[n] ⊆ N_state) then
      L[n] := 'S';
      N' := N' − {n};
    else
      // n is an element of the ND-block
      if (D_in[n] ⊆ N_state) then
        L[n] := 'ND';
        N' := N' − {n};
      fi
    fi
  done

  // backward traversal to compute L
  // L[n] set of related input nodes
  E' := E ∩ (N' × N');
  F := N_out ∩ N';
  forall n ∈ F do L[n] := D_in[n] ∩ N'; done
  forall n ∈ N do c[n] := 0; done

  while ( F ≠ ∅ ) do
    forall n ∈ F do
      forall n' ∈ N' with (n',n) ∈ E' do
        c[n'] := c[n'] + 1;
        if c[n'] == d_out((N', E'), n') then
          L[n'] := ⋂_{(n',n'')∈ E'} L[n''];
          F := F ∪ {n'};
        fi
      done
      F := F − {n};
    done
  done

  // n and n' are in the same block
  // iff L[n] = L[n']
  return (L);
end
```

A DYNAMIC TYPE CHANGES MODEL FOR ADAPTIVE SIMULATION ENVIRONMENTS BASED ON VIRTUAL MACHINE META-MODEL

Cristina Mîndruță
OVIDIUS University of Constanța,
B.dl Mamaia no.124,
900257 Constanța, Romania
E-mail: cmandruta@univ-ovidius.ro

KEYWORDS
Meta-modeling, virtual machine, adaptive simulation environment.

ABSTRACT

The paper uses a virtual machine meta-model based on the theory of sets in order to define a specific service and a protocol for adaptive feature in simulation environments. The service is responsible to type system management and one of its functions is to allow dynamic type changes, feature required in an adaptive simulation environment. The service implements a reflective view of the computing platform and of the application running on it, in terms of types, resulted from the definitions in the meta-model of the virtual machine. In order to implement the dynamic type change function of the service, a protocol, which defines both the relations between the specific service and the other services of the computing platform and also the change discipline in the type system of the execution environment, is defined. The proposed model offers adaptive feature on two levels for simulation environments, allowing changes not only at the application level, which gives the adaptive feature to the simulation itself, but also at the underlying execution support level, which gives the adaptive feature to the execution environment.

INTRODUCTION

Simulation environments are, generally, dedicated computing platforms. Different research projects are oriented to flexibility improvement.

Flexible simulation frameworks are designed based on the researcher expertise in modeling the simulated universe, identifying its main entities, devising an abstract model and introducing tuning parameters. Flexibility results from the parameter configuration. It may also be achieved by recursive executions based on recursive definitions of the entities in the universe (Albowicz et al. 2001).

Flexibility is also achieved by component-based architectures in which global data repository permits both actors running in the simulator and external analysis components to generate and use arbitrary pieces of information (Horling and Lesser 2004).

Object and component oriented architectures allow flexible systems design based on modularity. Modular simulation environments can be configured on a project-by-project basis or even dynamically during the course of the project using an object management based architecture as in (Standridge 1999). Another example of flexible simulation environment is (Vachharajani et al. 2003) which, based on component reuse, easy timing-control specification, a library of components and a set of tools, is capable of modeling any synchronous hardware system and of generating correspondent simulators.

Also for flexibility purposes, in (Pal et al. 2004) is addressed the issue of data schema evolution in integrated simulation environments from the perspective of incorporating different tools with these environments.

Our approach to flexibility is more abstract and general and proposes a model that offers the ability to monitor and control both the application and the infrastructure, and makes it possible to introduce the adaptive behavior. This is based on a reflective view of the type system of the simulation environment, which uses a formal meta-model that gives a representation of the type system of the virtual machine concept based on the theory of sets. The type system contains the interface types, the implementation types and the services types and also the correspondences between them.

This meta-model (Mîndruță 2004) is the model of a generic computing system seen from the virtual machine concept point of view. We consider the virtual machine as being the "atomic" computing system. Complex computing systems are built from simple ones by combination.

At a high level of abstraction, a virtual machine is an assembly of two main components: API (application programming interface) and runtime environment. This definition was refined and developed, in terms of types, giving us a glass-box view of the virtual machine. API is the set of function types and data types used in the applications developed for the virtual machine. The runtime environment is the set of services that support the applications execution.

The aim of this paper is to device a protocol that defines both the relations between a specific service of the execution environment and the other services of the computing platform and also the change discipline in the type system of the execution environment.

VIRTUAL MACHINE STATIC META-MODEL

In this section is presented the static part of the virtual machine meta-model, which contains interface types, implementation types and their relations.

Interface model

From its functionality point of view, a virtual machine is described by its application programming interface (API).

The interface of the virtual machine is formalized by the set of sets:

$$I_E = \{S, F, TE, AE\} \quad (1)$$

and by the set of correspondences:

$$I_C = \{fct, evt\}, \quad (2)$$

where
S is a finite set of types, representing the types of information (data) recognized by the virtual machine,
F is a finite set of types, representing the types of the functions that can be requested, by the application, to the virtual machine, which we call primitive functions,
TE is a finite set of types, representing the event types that can be generated by the virtual machine,
AE is a finite set of types, representing the external events that can be received by the virtual machine.

We use the definition in (Broy 1995) for the function

$$fct : F \rightarrow S^* \quad (3)$$

which associates its arguments types and result type sequence to each primitive function,
i.e. $\forall f \in F$, is defined $fct(f) = (s_1, \ldots, s_{n+1})$, with $s_1, \ldots, s_{n+1} \in S$.
Supplementary, we define the function

$$evt : F \rightarrow \mathcal{P}(TE) \quad (4)$$

which associates its generated event types set to each primitive function,
i.e. $\forall f \in F$, is defined $evt(f) = (t_1, \ldots, t_k)$, with $t_1, \ldots, t_k \in TE$.

Implementation model

Data types implementation model
Let V be a finite set of carrier data types, sources of types in S.

The function :

$$Srs: S \rightarrow \mathcal{P}(V)$$

associates the carrier set $s^V \in \mathcal{P}(V)$ to each $s \in S$, where $s^V = \{\delta_1, \delta_2, \ldots \delta_p\}$ is the set of data types, $\delta_j \in V$, used to create the definition of the type s.
At any moment, a data of type s is generated according to the composition of the definitions in the set s^V.
Function types implementation
Let P be a finite set of carrier function types. The function

$$Intp: F \rightarrow \mathcal{P}(P)$$

associates the set $f^P \in \mathcal{P}(P)$ to each $f \in F$. The set f^P is the set of functions whose composition, actually an algorithm, represents an implementation of f in a language defined over the alphabet P.

The sets P and V are related by a function similar to fct defined in (3):

$$fct^* : P \rightarrow V^*$$

Event types implementation
Synchronous events
In our approach, the internal events of a virtual machine, also called exceptions, are generated based on a logical predicates set implemented at the platform level and result from the current activity on the virtual machine.

Let SE be the set of internal events types.
The function

$$Trigg: SE \rightarrow \mathcal{P}(S) \times \mathcal{P}(P)$$

associates the sets $e^S \in \mathcal{P}(S)$ and $e^P \in \mathcal{P}(P)$ to each event type $e \in SE$.
The set e^S is the set of data types implied in the expression representing the event e triggering condition, expression, which we call predicate.
The set e^P is the set of function types whose composition (based on an algorithm) represents the implementation, in the language defined over the alphabet P, of the trigger actions, for the internal event of type e.

The sets P and SE are related by a function similar to evt defined in (4):

$$evt^* : P \rightarrow \mathcal{P}(SE)$$

This function associates to each carrier function in P, the set of internal events that its execution can generate. The types of these events are a subset of SE and their corresponding values for $Trigg$ are in the range $\varnothing \times P$.

Asynchronous events
Asynchronous events (interrupts) are generated by entities in the virtual machine's environment and are not synchronous with the current activity on the virtual machine. In complex systems some of the entities that generate asynchronous events can be virtual machines too.

As specified before, ***AE*** is the set of external event types and is a component part of the virtual machine interface.
The function :

$$Rut : AE \rightarrow \mathcal{P}(P)$$

associates the set $i^P \in \mathcal{P}(P)$ to each event type $i \in AE$.

Relations between event types
There are some important relations between event types.
The intersection of the sets ***SE*** and ***AE*** is empty.
Events are captured and locally handled by the current virtual machine, resulting in the virtual machine state changes. They may also generate events thrown to higher levels, as types in the set ***TE***, either as the same event type or as a refined event type. The union of the sets ***SE*** and ***AE, SE∪AE,*** contains a subset of types from which each type generates a subset of types in ***TE.*** In order to model this, we define the function

$$Th: SE \cup AE \rightarrow \mathcal{P}(TE)$$

which associates the set $t^e \in \mathcal{P}(TE)$ to each $e \in SE$ or the set $t^i \in \mathcal{P}(TE)$ to each $i \in AE$. The sets t^e and t^i are the sets of events generated in the interface of the virtual machine by the event ***e*** and ***i*** respectively. We may say that the set ***TE*** contains refinements of a subset of the event types set ***SE∪AE.***

Virtual machine static formal meta-model

We conclude that the static meta-model of the virtual machine is defined by the following set of sets:

$$VM^E = \{ S, F, TE , V, P, SE, AE\} \quad (5)$$

and by the following set of correspondences:

$$VM^C = \{fct, evt, Srs, Intp, Trigg, Rut, Th\} \quad (6)$$

Dynamic meta-model

Runtime core services model

In this section is formalized, in terms of types, the dynamic aspect of the virtual machine. From the type system point of view, the dynamic aspect of the virtual machine refers to the service types, which offer a collection of virtual resources supporting applications execution.

The fundamental virtual resources in computing systems are a virtual processor, a virtual memory and a virtual communication space with the environment. These abstractions are realized by specific services and mechanisms. According to the fundamental models of computing systems, we consider that the core services are: processing service, memory service, event service and predicate service. The service collection represents the support for the execution of programs written in the virtual machine language.

The dynamic meta-model includes these services in relation to the static meta-model.

Processing service (***PSv***) is responsible to process the functions in the set ***F***. For that it uses representations of the sets ***S, F*** and ***P*** and of the correspondences ***fct*** and ***Intp***.
Memory service (***MSv***) offers storage functions for the data according to data types in the set ***S***. It uses representations of the sets ***S, V*** and of the correspondences ***Srs***. Specific management of these sets allows different memory models implementation.
Event service (***ESv***) is responsible to external events management. It uses representations of the sets ***AE, TE, P*** and of the correspondences ***Rut, Th.***
Predicate service (***TSv***) implements internal events (exceptions) mechanism. For that it uses representations of the sets ***S, F, SE, TE, P*** and the correspondences ***evt, Trigg, Th***.

Sv is the set of services. The correspondence:

$$dy: Sv \rightarrow \mathcal{P}(VM^E) \times \mathcal{P}(VM^C) \quad (7)$$

associates to each service $sv \in Sv$ the corresponding sets of entities in the static meta-model.
According to this, the values of the function ***dy*** applied to the core services are:

$$dy(PSv) = \{\{S, F, P\},\{fct, Intp\}\} \quad (8)$$

$$dy(MSv) = \{\{S,V\},\{Srs\}\} \quad (9)$$

$$dy(ESv) = \{\{AE, TE, P\},\{Rut, Th\big|_{AE}\}\} \quad (10)$$

$$dy(TSv) = \{\{S, F, SE, TE, P\},\{evt, Trigg, Th\big|_{SE}\}\} \quad (11)$$

The complete formal meta-model of the virtual machine in terms of types is:

$$VM = \{VM^E, VM^C, Sv, dy\} \quad (12)$$

Reflective virtual machines

The core meta-model of the virtual machine, containing the fundamental set of service types, is extended with a new service, which is the manager of the virtual machine type system. This service is the component that adds the reflective feature to the virtual machine.

Core services system of the virtual machine implements classic type management functions (ex. runtime type validation, runtime type instantiation from a predefined set of types). The new service is based on the separation of type management functions from the core service system functions. The responsibility of classic type management is removed from each of the core services and handled over to the new service. This separation increases the flexibility of the system due to the benefits of modularization and independence. It also allows us to extend the new service with new type management functions.
The new function we consider is runtime type definition, which means the possibility to manage dynamic changes in the type system of the virtual machine. Being a runtime service of the virtual machine, it is the basis for runtime reflection.

By separating and extending the type management functions, we will obtain a flexible virtual machine with

two distinct functions: the function of modeling and dynamic building of application and of virtual machine components and the classic function of application execution support. According to this, the dynamic meta-model is extended with a new service as previously discussed. This service is responsible to type creation and instantiation. We call it *modeling and dynamic building service* and denote it by ***DSv***.

The *modeling and dynamic building service* benefits from runtime access to the sets of the static meta-model, which contains the static components of the virtual machine type system. The service manages the sets of types ***S, F, TE, SE, AE*** and offers operations for creation and for elimination of types in the system. The service is also responsible to entities dynamic building based on these type definitions. The building operation uses carrier sets ***V*** and ***P*** and updates correspondences in the set VM^C. In conclusion, the value of the ***dy*** function for this service is:

$$dy(DSv) = \{\{ S, F, TE, V, P, SE, AE \}, \{fct, evt, Srs, Intp, Trigg, Rut, Th\}\}. \quad (13)$$

It is obvious that $dy(DSv) = \{VM^E, VM^C\}$, which justifies the necessity of all the elements included in our static model for the purpose of runtime type management.

We conclude that the dynamic meta-model of the virtual machine with flexible type management is represented by the set of services:

$$Sv = \{PSv, MSv, ESv, TSv, DSv\} \quad (14)$$

and by the correspondence ***dy*** defined in (7), whose values were given in (8), (9), (10), (11) and (13).

INTER-SERVICES COMMUNICATION PROTOCOL

Static type system graph representation

The static meta-model of the virtual machine in (5) and (6) can also be represented as the following graph, in which a node is a type set and an oriented arc is the correspondence defined between the two type sets.

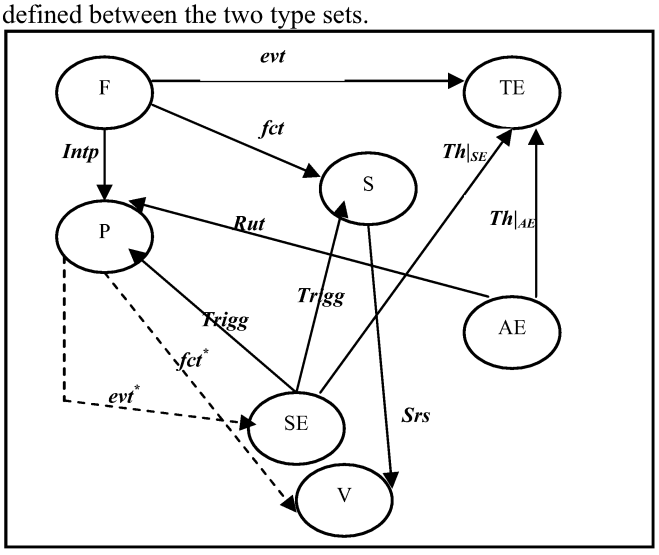


Fig.1 Static Virtual Machine Type System Graph Representation

Dynamic changes type management

Analyzing and processing the previous graph, we can identify dynamic type management sequences.
Independent nodes from the type change point of view are the ones that have no output arcs. So, any type change in the sets ***V*** and ***TE*** does not have any consequences as regards changes management for any other type or correspondence of the model. In other words, a change in ***V*** or in ***TE*** is transparent to the rest of the elements of the model. When it is done it is straightforward applied to the whole system without any other modification. We consider these sets to be at the first level of change independence.

In the same category is a subset of ***SE*** which is the union of the elements in the sets that are the components of $Im(evt^*)$. This subset represents the internal events that can be generated by the execution of carrier functions themselves and we denote it by SE_u.

Exploring the graph we find the sets that have an output to ***V***, to ***TE*** or to SE_u. The second level of change independence is represented by the sets that have outputs only to sets on the first level of changes independences. On this level there are the sets ***S*** and ***P***. Any change in the set ***S*** can be done only after correspondent changes in the set ***V***, if they exist (a change in the set ***S*** may use only unchanged elements in the set ***V***) but always will imply a change in the ***Srs*** correspondence. The similar considerations for the set ***P*** apply to the sets ***V*** and SE_u and the correspondences fct^* and evt^*.

We have defined a generic recursive function *Change(type set, changed_item)* that returns „false" if the change is not needed or else realizes the type set change and returns „true". It applies straightforward for changes of the elements in the sets on the first level of independence:

Change(V, changed_item)
Change(TE, changed_item)

The functions *Change* for elements in the sets on the second level of independence are:

Change(S,changed_item)
    if (Change(V, changed_item)) then Update(Srs)

Change(P,changed_item)
    if (Change(V, changed_item)) then Update(fct^*)
    if (Change(SE_u, changed_item)) then Update(evt^*)

where *Update(corresp_type)* is the function that updates the correspondence specified as a parameter.

Exploring the graph in the same manner towards lower independence levels, the change functions corresponding to all the nodes in the graph have been defined.

Inter-services relations for adaptive feature

The *modeling and dynamic building service* is responsible to type management operations in the execution environment. One of its functions is dynamic type change, the basis for the adaptive feature of the runtime environment.
Dynamic type modifications must be synchronized with the type current usage in the system. This can be realized by introducing a new mechanism that must implement an inter-services communication protocol in order to negotiate the type modification.

Negotiation with a service is a relation between the *modeling and dynamic building service* and the other service of the execution environment. This operation is executed at runtime in order to allow dynamic integration of types as support for the adaptive feature. Negotiation can be more or less complex. In the simplest case it means that *modeling and dynamic building service* asks the other service if the type is in use and waits until the service returns a negative answer. When the type to be changed is not in use any more, the change can be integrated in the platform.

The selection of the services to negotiate with about the type to be changed is done according to the values of the function ***dy*** defined in (8), (9), (10), (11) and (13).

The independence levels resulted from the graph in fig. 1 are represented in the following table:

level	set	correspondence	to set
1	V, TE, SE_u	-	-
2	S	Srs	V
	P	fct^*	V
		evt^*	SE_u
	F	evt	TE
	$SE \setminus SE_u$	$Th\|_{SE}$	TE
	AE	$Th\|_{AE}$	TE
3	F	fct	S
		$Intp$	P
	$SE \setminus SE_u$	$Trigg$	S, P
	AE	Rut	P

The table is useful in order to define the type change discipline. A type change is done according to a discipline based on the following rules:

1. A change of a type begins with the necessary additions on the smallest independence levels and propagates through the graph to the current level, updating all the correspondences and the type sets in the path.

2. A change of a type reflects in its type set and in correspondences on its output arcs. It is negotiated with all the services that have the representations of the specified set and correspondences.

This discipline is implemented by *modeling and dynamic building* service in order to manage dynamic type changes requested by adaptive execution environments.

CONCLUSIONS

The formal meta-model gives a representation of the type system of the virtual machine concept based on the theory of sets. The type system containes the interface types, the implementation types and the services types and also the correspondences between them.

The formal meta-model of the virtual machine offers a unified and sound view of computing platforms in terms of types. Its runtime availability allows dynamic changes in the type specifications of the components of the virtual machine. A reflective virtual machine is built by extending the core services model with a new service responsible to type system management, service which benefits of a reflective view based on the definitions in the formal meta-model.

Using the definitions in the formal meta-model, the paper identifies the elements of a protocol for dynamic changes in the type system, requested in an adaptive execution environment. The protocol defines the relations, used in a negotiation mechanism, between the *modeling and dynamic building service* and the other services of the platform and also the change discipline in the type system of the execution environment.

The proposed adaptive virtual machine is a model for simulation environments that are adaptive on two levels, because it allows changes not only at the application level, which gives the adaptive feature to the simulation itself, but also at the underlying execution support level, which gives the adaptive feature to the execution environment.

REFERENCES

Albowicz, J.; Chen A.; and Zhang L., “Recursive Position Estimation in Sensor Networks”. In *IEEE Conf. On Network Protocols*, Nov. 2001, 35-41.

Horling, B.; and Lesser,V., “Data Dissemination Techniques For Distributed Simulation Environments”, *in Proceedings of the Winter Simulation Conference, WSC 2004,* dec. 2004

Standridge, C., “Modular Simulation Environments”. In *Proceedings of the 1999 Winter Simulation Conference*, P.A.Farrington, H.B.Nembhard, D.T.Sturrock, and G.W.Evans (Eds.), , 598-602.

Vachharajani, M.; Vachharajani, N.; Penry, D,; Blome, J.; and August, D., “The Liberty Simulation Environment, Version 1.0”.

Pal, V.; Papamichael, K.; Bourassa, N.; and Loffeld, J., “Incorporating LCA tools in integrated simulation environments”. In *Seventh International IBPSA Conference,* Rio de Janeiro, Brazil, August 13-15, 2001, 689-696.

Mîndruță, C., ″A Formal Model For Flexible Type Management On Virtual Machines″, unpublished.

Broy, M., ″Mathematical System Models as a Basis of Software Engineering″. In *Computer Science Today 1995*.

A METHODOLOGY TO REDUCE THE DESIGN LIFECYCLE OF REAL-TIME EMBEDDED CONTROL SYSTEMS

Rémy KOCIK
ESIEE, Cité Descartes - BP 99 - 2, BD Blaise-Pascal
93162 Noisy-le-Grand CEDEX – France
E-Mail : kocikr@esiee.fr

Yves SOREL
INRIA Rocquencourt
78152 Le Chesnay CEDEX
E-Mail : yves.sorel@inria.fr

KEYWORDS

Control Systems, Real-time, Multiprocessors, Hybrid Simulation, Computer Aided Design

ABSTRACT

There are two gaps in the typical design V-process used for real-time embedded control systems. The first one is due to the manual translation of control laws into a software specification; the second one is due to the manual implementation of this specification onto distributed architecture while satisfying real-time constraints. Indeed, the correctness and efficiency of these two translations mainly rely on the skill of engineers who are subject to errors, the latter requiring many design process iterations to be corrected. We propose to fill the first gap between the modeling and the specification phases, firstly by linking the terminologies used by the two communities involved, and secondly by interfacing Scicos a hybrid simulation tool with SynDEx a specification and optimized implementation tool. The second gap is actually filled by SynDEx itself which assists the designer in implementing the control laws onto distributed architecture with the help of optimization heuristics and automatic code generation. We illustrate the proposed approach through a didactic example of inverse pendulum control system.

CONTROL EMBEDDED SYSTEMS DESIGN LIFECYCLE

Designing control embedded systems requires to solve many hardware and software problems: complex application algorithms must be implemented onto heterogeneous distributed architectures composed of processors (DSP, RISC) and specific integrated circuits (ASIC, FPGA) all together interconnected. In order to deal with this complexity, methods are often based on hierarchical design allowing to describe the system as a set of simplest, and easiest to design sub-systems. These methods are applied in the typical development lifecycle called ``V-process''. It allows to build a system starting from an abstract description to an actual product which is validated by a step by step top-down and then bottom-up design flow (Calvez 1993).

The top-down design is decomposed in three main phases: modeling, specification and implementation. The bottom-up design is decomposed in three validation phases: unit test, integration test and validation test of the system (see Figure 1). This V-process can be used both for hardware and software designs. In this paper, we focus on the software design.

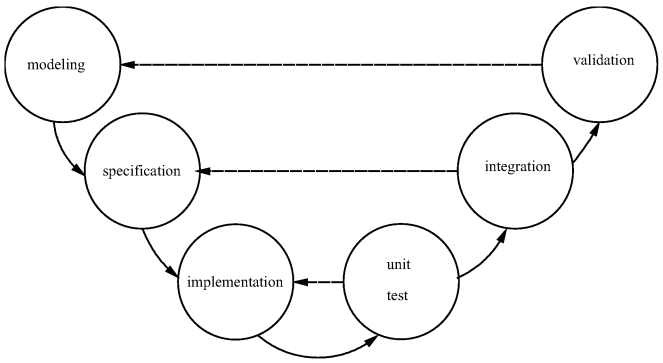


Figure 1: V-Process for Embedded Control System Design

Modeling corresponds to the mathematical description of the behaviour the system must conform to. Software *specification* consists in a high level description of the algorithms which implement the mathematical equations resulting from the modeling phase, and a description of the implementation constraints, i.e. real-time and hardware constraints. The software is actually written in the *implementation* phase in order to ensure that the algorithms will be computed on the architecture meeting the implementation constraints.

The first validation phase in the development process is the *unit test* which is often long and tedious. It consists in verifying that each software function implementing the algorithms are correctly written and perform right. When all the functions are debugged and separately validated, the *integration* phase verifies that the set of all the functions behave as defined by the model. If not, specification has to be modified. The last phase, *validation*, consists in verifying that the application behaviour is consistent with the requirements (customer's specification). If not, the model has to be modified, and usually refined. In this case the V-process is iterated as long as necessary.

GAP IN THE V-PROCESS

In the development process of a control system, control engineers define the system model and computer science engineers are in charge of the specification and implementation part. Both have to collaborate in the specification phase. This is an important phase where the mathematical model is transformed in a computing model. Problems involved by these two models are different and

errors due to misunderstanding between control engineers and computer science engineers may appear in this transformation. These errors may have impacts up to the implementation phase and then be only detected during the validation phase. Thus, development lifecycle time may be strongly increased due to these numerous backtrackings.

In this paper, we first describe, using control engineer terminology the modeling phase based on control terminology. Then, we explain using real-time computing terminology the implementation phase of the control models. Finally the links between both terminologies are made in order to reduce the gap between these two communities.

The V-process has shown a good efficiency in the design of many applications. But their growing complexity leads to an important development cost rise which became an important part of the final product cost. Thus, time spent by engineers in writing and debugging software code, is the most important part in the design process. In order to ensure competitiveness of products, it is necessary to minimize the software development lifecycle by reducing the duration of each phase, and by minimizing the number of iterations in the global V-process.

Tools based on formal languages (Halbwachs 1993) are intended to improve the global process. Indeed, these languages rely on rigorous specification based on mathematical rules. This makes possible specification, verifications, and automatic code generation such that the number of iterations in the V-process is reduced because verifications allow to detect more early specification errors. The automatic generation of a code consistent with the verified specification allows also to reduce tests and debug.

In this context, the AAA methodology (Algorithm Architecture Adequation) based on a graph formalism as been developed to optimize the implementation of application algorithms onto distributed architectures while satisfying real-time constraints (Sorel 1994). The system level CAD tool SynDEx (`www.syndex.org`) (Grandpierre and al. 1999) which relies on this methodology allows to quickly develop complex applications such that, for example, an automatic driving application for the CyCab an electrical vehicle based on a distributed architecture involving several MPC555 microcontrollers interconnected through a CAN bus (Kocik and Sorel 1998). This experience has shown the benefits that this methodology can bring for the design of complex real-time applications.

Nevertheless, this methodology takes into account only the specification and implementation part of the V-process, and there is a gap between the modeling and specification phases which may introduce errors very early in the development process. In industry this translation from modeling to specification and implementation usually relies on the skill of only few peoples. It is necessary to be vigilant because errors introduced at this level may have consequences along the development process, and are often only detected during the validation phase. Thus, correcting the error imposes a complete iteration of the development process.

Most errors usually occur during the controller model design, these errors being introduced by the translation of this model into its software specification. In order to reduce errors, to enhance ``traceability", and to minimize the number of development process iterations we propose to automatically translate models into software specifications. In the last part of this paper, we show how interfacing Scicos (`www.scicos.org`), a hybrid dynamic systems modeler and simulator, with SynDEx allows to achieve this goal.

CONTROL SYSTEM MODELING

The job of control engineers is to build a physical system able to keep under control the evolution of an other physical system called *plant*. The latter may possibly be composed of mechanical and/or electrical and/or chemical components. First the control engineer has to describe with mathematical equations the interactions between these components. According to this mathematical model of plant, he can predict its evolution when it is submitted to external control actions. Then, he can give a mathematical model, called *control law*, of control actions to be applied to the plant. A controller which purpose is to physically perform the control laws, has to be build and to be connected to the plant in order to control it. The plant (system whose we want to control the evolution) and the controller (system which has to act on the plant) linked together, make a system that is called *control system* (Figure 2).

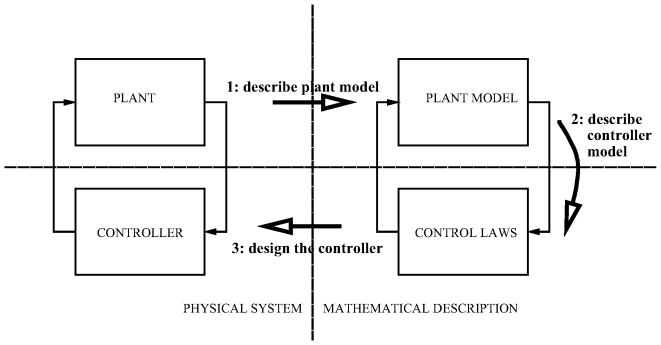


Figure 2: Control System Design

A complex controller can be designed from the composition of several basic control laws such as single-input-single-output controller. Its input is called the *reference* that the plant as to reach. Its output is called *control* which is applied to the plant in order to ensure that the plant evolves towards the reference.

Control systems are usually *loopback systems*. Thanks to a permanent observation of the plant evolution through a *feedback* issued from the output of the plant, the control system is able to remove external *disturbance* (Ogata 1970).

DIGITAL CONTROLLER

Sampled Control Law

When the plant is complex, it can be necessary to use a computer to perform in real-time the control laws producing actions applied to it. This solution may reduce the costs, and

moreover increase the performances of the system (Ogata 1987).

The implementation of a control law by a computer system requires to discretize input and output of the plant (Landau 1988). Thus, interactions between the computer and the plant are made using *transductors*. A feedback signal is produced by a *sensor*, which measures the amplitude of physical phenomenon and translates it into an electrical signal sampled by an analog to digital converter (ADC). The computed output is a digital signal which is converted to an electrical analog signal by a digital to analog converter (DAC) before being applied to an *actuator*. Its role consist in converting the electrical signal into an other physical phenomenon able to control the plant evolution (Figure 3) (Astrom and Wittenmark 1984).

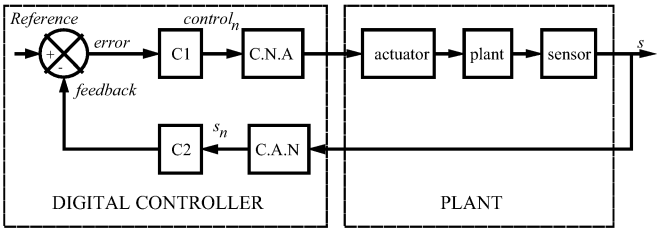


Figure 3: Computer Loopback System

The control law supplying that output signal from feedback and reference inputs is described by an algorithm implemented in a program executed on a computer. The computer seen, by the control engineer, as a simple element of the controller, is the center of interest of computer science engineers. From his point of view, the computer science engineer has to design an *application* made with a *computer system* and a physical *environment* both interacting. This system is composed of a *computer* and a set of programs, called *software*, that it will execute. The environment is defined as the set of all the physical elements which are external to the computer system. The frontier between the environment and the computer system is often difficult to establish. This is the reason why we choose here that all the physical components which can be programmed (processors, network, memory, input/output devices, ...) are parts of the computer system, whereas all the physical components which cannot be programmed are parts of the environment. The latter components, except the ADC and DAC ones, are seen by control engineers as the plant.

The computer system has to ensure that the plant behaves according to the control laws defined by the control engineer in the modeling phase. It has to interact permanently with its environment: the feedback and reference sampled input values are *input events* on which computations are performed producing *output events* which are discrete values. That is the reason why this kind of computer system is also called *reactive system* (Harel and Pnueli 1985).

The correctness of an actuation usually depends on the time elapsed between the generation of an output event and the input event which has triggered it. In this kind of systems, this time interval have to be bounded. Such applications are called *real-time applications*, the time requirements are called *real-time constraints*. In some applications, it is possible to accept sometimes that some real-time constraints are not always met, but usually these requirements are critical and have to be imperatively met. The main difficulty is to design a *predictable* real-time system guaranteeing that all the critical real-time constraints are always satisfied and the overtaking of non-critical real-time constraints will remain bounded, and casual (Le Lann 1990; Stankovic and Ramamritham 1993).

Specification

Specification corresponds to a high level software and hardware description (Calvez 1993). In the case of real-time systems these two aspects are strongly linked. In order to design the software, it is needed to specify the algorithms to be computed, the hardware architecture which will execute the algorithms, and the real-time constraints that must be satisfied (Mathai 1996).

Real-time Constraints

The discretization and digital implementation of control laws have some consequences on modeling. The control engineer has to choose sampling frequencies for each input and each output of the system (*control, s* and *reference* in Figure 3). As a general rule, in order to simplify complex computations induced by control laws discretization, the control engineer assumes that all the signals are periodically sampled. For the same reason, he assumes that input and output signals are sampled at the same time instant (Ogata 1987). Control theory books shows that this choice is empirical and relies on the skill of control engineers. The sampling frequency choice is translated during the implementation by a constraint imposed to the input events rate that the real-time system may accept. This constraint is called *input rate*. A *latency* constraint is also imposed to the real-time system. It is a boundary on the time interval between an input event arrival and the production of the corresponding output event. The goal of this constraint is to guarantee the *response time* of the system, an important quality criteria from the control engineer point of view. It has been shown, under simplifying hypothesis, that this response time is proportional to the computation latency added with a delay $T_e\backslash 2$ due to sampling performed at $1\backslash T_e$ frequency (Phillips and Troy 1984).

Algorithm Specification

Taking into account the infinite iteration due to the interactions between the real-time system and its environment (the number of iterations cannot be bounded, considered as infinite) the typical definition of an a*lgorithm* extended such as an infinite sequence of operations computed in a finite time on a finite hardware. In this way, an algorithm is seen as a finite sequence of operations computed in a finite time on a finite hardware, but infinitely repeated. The execution of every sequence of operations is triggered by input events which may be periodic or aperiodic. Complex real-time systems involve two kinds of algorithms: *data processing* and *state machine*. Data processing algorithms describe control actions applied to the plant. They represent the computations the control laws perform on data (PID corrector, filter,...) considered as periodic events

because they are sampled at the input rate. State machine algorithms define in which order data processing algorithms must be executed according to the current state and aperiodic input events.

Hardware Specification
Embedded applications are subject to strong cost constraints (financial cost, dimensions, electrical consumption). To meet these constraints, new hardware architectures have been proposed. For example, in automotive industry where competitivity is particularly hard these new technologies have contributed to reduce the amount of wiring thanks to the integration of sensors and actuators near to the processor, and to data multiplexing on serial bus. This lead to heterogenous distributed architectures with low cost components of the shelf. Computers are build with microcontrollers to perform evolving functions, and with ASICS, and/or FPGA to perform some specific functions with only few evolutions in the product life. Communications between processors are supported by low cost serial buses (like VAN, CAN, TTP, FlexRay,…) well suited for disturbed environment.

DESIGN LIFECYCLE REDUCING

In order to meet the implementation constraints and to avoid gaps between all the V-process phases and thus reduce design costs we propose a methodology based on the cooperation of two tools: Scicos for modeling and hybrid simulation, and SynDEx for specification and distributed implementation.

Specification

In computer control systems sampling and quantifying operations, needed for the discretization of analog signals, may introduce some errors which tend to degrade system performances. For the same performance level, if we want to take into account these errors, it is more complex to define the control law for a computer system than for an analog one. The modeling of a discretized system is more complex than the modeling of a continuous system for which control theory brings numerous mathematical tools.

In order to simplify the design, the control engineer usually studies a continuous model of the controller rather than a discrete one (Astrom and Wittenmark 1984). Then, this continuous model is discretized in order to allow its implementation on a computer. Thus, an hybrid simulation, i.e. the simulation of a continuous system (the plant) linked with a discrete real-time system is mandatory to validate the model used to design the real-time system. This simulation allows to tune the control laws in order to take into account approximations done in the continuous model.
Scicos (Nikoukha and Steer 1999) has been designed for this purpose, it allows an hybrid simulation taking into account the sampling frequencies of analog signals.

Specification and Optimized Algorithms Implementation

Discretized control laws and continuous plant models are described in Scicos with graphs similar to block diagram well known by control engineers. Such a graph representing a discretized control laws is extracted from Scicos and translated into a SynDEx graph (Djenidi and al. 1999). In this way, the algorithm specification conforms to the model, it is still not necessary to spent any time for the control laws specification, and only hardware architecture description is now required. SynDEx allows to perform this hardware specification, and when both algorithm (control laws) and architecture are specified it allows to execute the adequation, that is to say to execute heuristics which optimize the implementation of the algorithm onto the architecture and automatically generates the corresponding code that will be executed in real-time. Actually, interfacing Scicos with SynDEx allows to cover the complete design process.

Hybrid Simulation with Delays

Hybrid simulation, to be close to reality, needs to take into account delays introduced in the control laws by computation and communication durations (latency). These delays may impact on system stability(Torngren 1990). They depend on the implementation, that is to say on the allocation (distribution) and the scheduling of functions, associated to each block of the graph describing the control laws, onto processors.

Usually, in the specification phase each block is translated into a *task*, function with real-time execution constraints and properties (execution periods, priority, deadlines,…). Then, the schedule of this set of task is made at execution time (on-line scheduling) by an RTOS (real-time operating system) according to the properties of the tasks (Timmerman 1999). In this approach it is difficult to predict computing latency, and consequently it is not possible to easily take into account delays in the hybrid simulation. This problem can be solved by interfacing Scicos and SynDEx. Indeed, SynDEx is able to compute an adequation which is an optimized distribution (spatial allocation of CPU resources) and scheduling of the blocks (called in this context *operations*) of the algorithm onto the hardware architecture. Thanks to a good knowledge of the architecture (number of programmable and non programmable components, number of communication media) and to the knowledge of the maximum execution duration of each operation depending on the component able to execute it, SynDEx can compute and visualize a prediction of the optimized distribution and scheduling. This accurate prediction allows to verify whether real-time constraints are met or not, and also allows to provide back, in the hybrid simulation, the delays due to computations and communications. Thus, it is possible to simulate the hybrid system with Scicos in order to verify that response time, stability and others requirements are met in spite of these delays.

Example

In order to show the benefits of this approach we illustrate it with a didactic example of an inverse pendulum control system. The inverse pendulum is a system composed of a cart (mass M) on which a bar (mass m, length l) is linked. The bar can rotate around its extremity. The cart stays on a

inclined plane. The goal is to stabilize the cart at the origin position *O* (Figure 4).

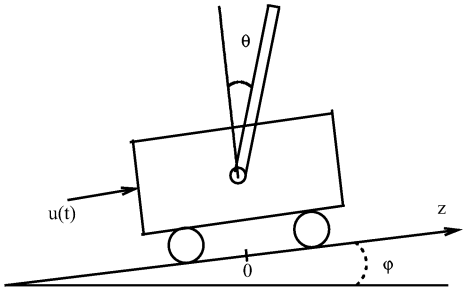


Figure 4: Inverse Pendulum Stabilization

We suppose that some sensors provide the angular position θ and the linear position *z* of the cart on the plane. The control action is a translation force *u(t)* produced by an electrical motor.

Modeling and Simulation

The first phase corresponds to the modeling and the simulation of the system in order to verify that the goal is met. Figure 5 shows how the modeling was performed with Scicos. The block named `cart` is a mathematical continuous model of the cart. This block is a function written in C or Fortran code and/or with Scicos library blocks. This model estimates *z* and θ from the angle φ (0.001 radians here) and from the control action *u(t)* applied to it. The `sensor` block models two sensors acquiring *z* and θ. The block named `actuator` is the electrical motor model. The `control` block is the discretized model of the controller. It describes the algorithms which will be executed in real-time.

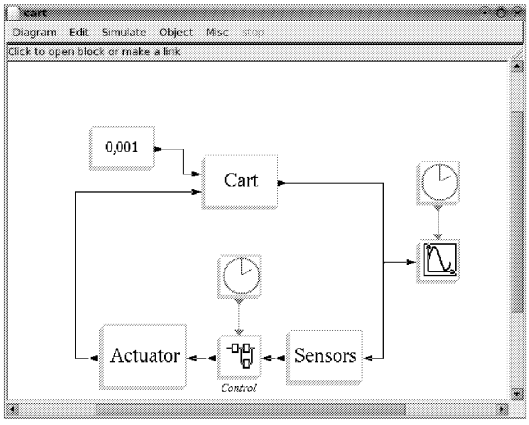


Figure 5: Scicos Graph

The sampling period of its inputs is 10ms, it is specified by a clock signal connected to its activation input port. The control engineer has used here a hierarchical description, the `control` block is a Scicos *super-block* which groups many C or Scicos blocks. Finally, a specific block (icon showing a 2D graph) allows to display a simulation graph of the *z* position and of the θ angle evolution.

Control Algorithms Extraction

When hybrid simulation shows that the control system is properly designed, the engineer may extract from Scicos the controller algorithms by selecting with the mouse the involved region (Figure 6).

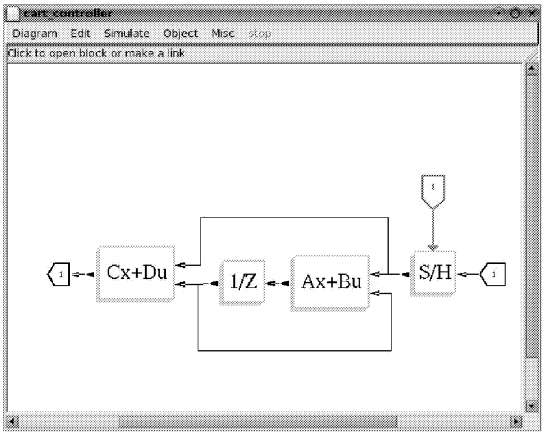


Figure 6: Controller Extraction

We can see that the control engineer chose here to design the control law as a matrix computation given by a state space representation. This Scicos graph region of the discretized control law can then be automatically translated into a SynDEx algorithm graph.

Real-time System Specification

The algorithm graph extracted from Scicos is translated in the SynDEx algorithm syntax. Now, the user must specify the architecture graph describing the hardware architecture of the real-time system and he must provide to SynDEx the worst execution duration of each operation of the algorithm graph. These durations can be, in a first step, estimated by engineers or they can be measured by SynDEx after a first execution of the automatically generated code.

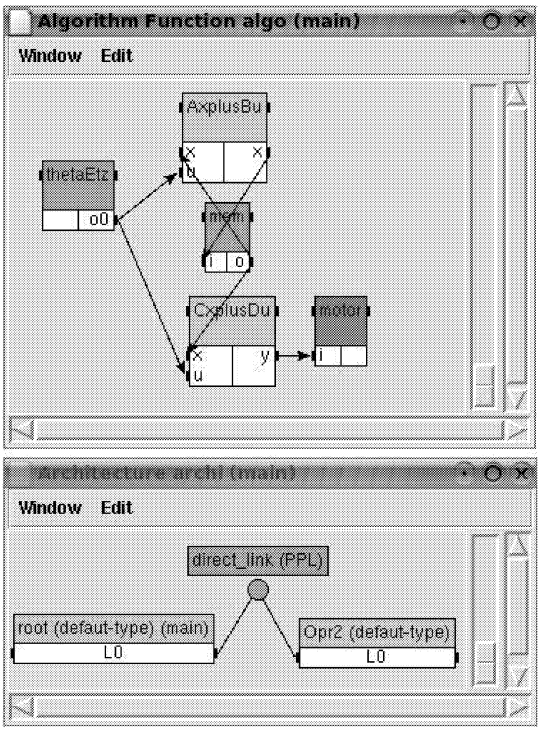


Figure 7: SynDEx Software and Hardware Graphs

The Figure 7 shows the SynDEx graphical interface where both algorithm and architecture graphs are displayed. On the algorithm graph (upper part), we can see the Scicos blocks `AxplusBu` and `CxplusDu` performing the matrix computation. The block called `thetaEtz` corresponds to the sampling of the inputs sensors while the block called `motor` drives the motor. The block `mem` specify the scicos block `1/Z`. For demonstration purpose, we have supposed that the hardware architecture is made of two processors linked together by a communication media called `direct_lnk` (Figure 7 lower part). Sensors are physically linked to the processor called `root`, the motor is physically linked through a power amplifier on the processor called `Opr2`.

Adequation and Temporal Simulation

From this specification, SynDEx can perform the adequation and display a temporal prediction of the algorithm execution onto the hardware architecture (Figure 8). Each column represents an execution sequence displaying one iteration of the operations distributed onto a processor, or of the data dependences distributed onto a communication media. In our case, there is one computing sequence on the `root` processor, another computing sequence on the processor `opr2` and a communication sequence on `direct_link`. The vertical axis represents the time evolution. It can be noticed that the complete execution of all the operations and all the communications needed is estimated to 3ms (3060). Thus, we can verify that the architecture can meet the temporal constraint of the input rate (10ms) given by the sampling period chosen for *z* and θ.

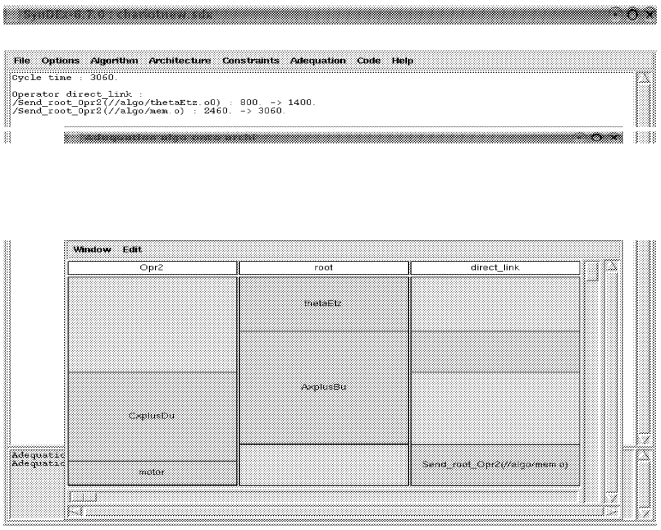


Figure 8: SynDEx Timing Simulation

Hybrid Simulation Taking into Account Delays

The computation latency (3ms) given by the SynDEx timing simulation is the delay between the sampling of *z* and θ and the *u(t)* output. This delay can be introduced in the Scicos graph in order to perform a more accurate hybrid simulation.

The Figure 9 shows the first simulation made without any delay. It exhibits that the control law discretized with a 10ms sampling period allows to stabilize the inverse-pendulum at *O*. Figure 11 and Figure 10 are the simulation results taking into account the 3ms delay. The first one (Figure 10) shows that with a 10ms sampling period, the system with the delay is no more stable. The second one (Figure 11) validates the control system stability with a control law discretized with a 8ms sampling period despite of the delay.

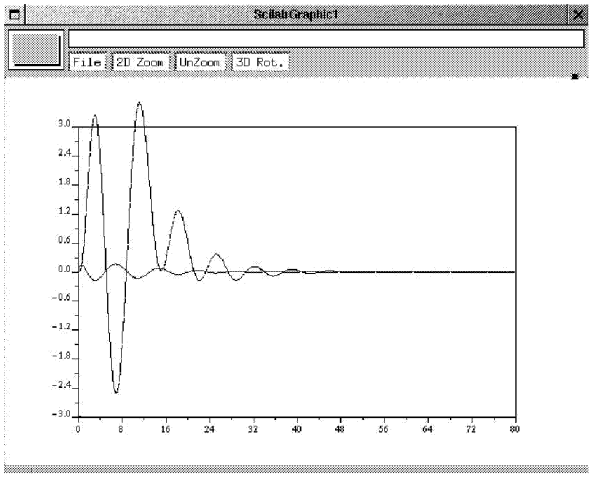


Figure 9: Simulation with 10ms Sampling Period, No Input/Output Delay

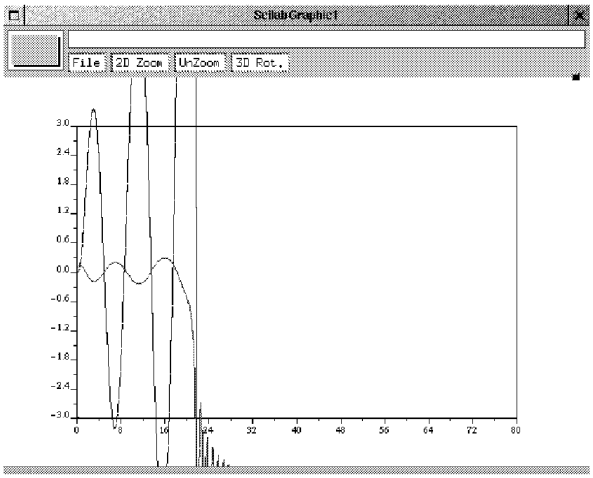


Figure 10: 10ms Sampling Period with 3ms Delay

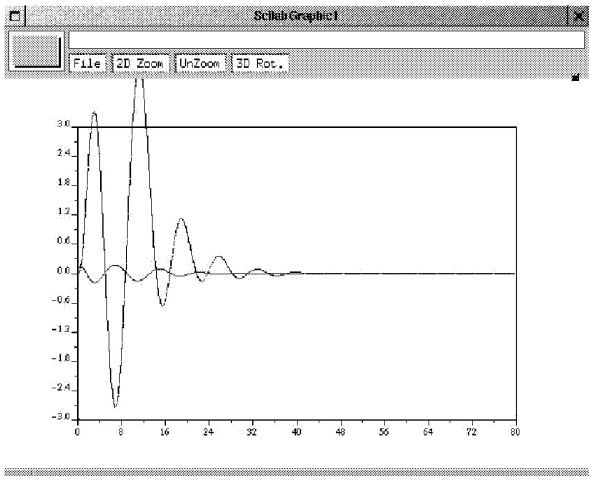


Figure 11: 8ms Sampling Period with 3ms Delay

Automatic Code Generation

When both hybrid (Scicos) and temporal (SynDEx) simulations show that the performances are satisfied, it is possible with SynDEx to generate the code that will actually be executed on each processors. SynDEx produces an executive involving inter processor communications which guarantees that the functions calls associated to each operation of the algorithm graph, and to each `SEND` and `RECEIVE` communication primitives, follows the computed schedule. The software engineer just has to provide the input and output functions (`thetaEtz` and `motor`). It is possible to reuse the C blocks used in Scicos to describe the control system: thus, the code that have been used for simulation is the same that the one used at execution time in the real-time system.

New Reduced Lifecycle

Figure 12 shows the new reduced lifecycle obtained using this methodology. Each phase is validated by simulation or verification before going to next the phase. Manual translation of models into specifications are minimized in order to avoid errors. We hope by this way to lead to a lifecycle closed to an ideal waterfall process without rise: each phase is validated before going to the next one.

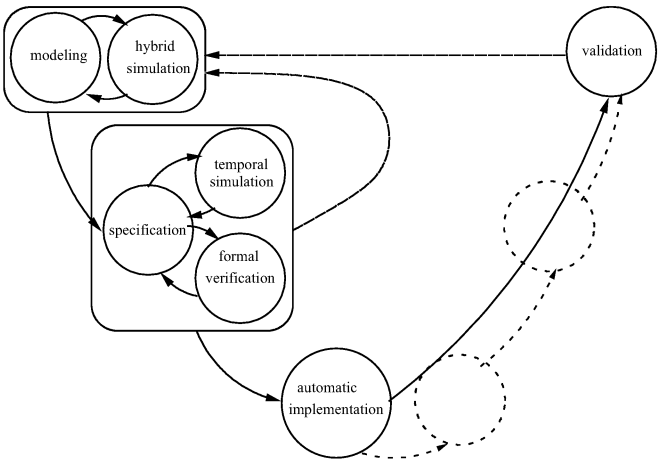


Figure 12: Reduced Design Lifecycle

CONCLUSION AND WORK IN PROGRESS

In this paper, we have described a methodology able to reduce the lifecycle of control embedded control systems. With a didactic example, we have shown that using Scicos for hybrid simulation taking into account the input-output delays predicted by SynDEx may reduce significantly the backtracking during the design V-process. Actually, the automatic code generation for heterogenous distributed architectures performed by SynDEx also decrease the design lifecycle. Presently, delays must be determined and introduced manually in Scicos. We are now working on SynDEx and Scicos tools in order to do this automatically. Morever, this new approach is actually evaluated on the design of a complex industrial automotive application.

Astrom, K.J. and Wittenmark, B. 1984. *Computer Controlled Systems: Theory and Design.* Prentice-Hall Inter-national.

Calvez, J. P. 1993. *A Specification And Design Methodology*. John Wiley Publisher.

Djenidi, R.; Lavarenne, C.; Nikoukha, R.; Sorel, Y.; and Steer, S. 1999. "From hybrid system simulation to real-time implementation". In *11th European Simulation Symposium and Exhibition* (Erlangen-Nuremberg, Oct.).

Grandpierre, T. ; Lavarenne, C. ; and Sorel, Y. 1999. "Optimized rapid prototyping for real time embedded heterogeneous multiprocessors". In *CODES'99 7th International Workshop on Hardware/Software Co-Design* (Rome).

Halbwachs, N. 1993. *Synchronous programming of reactive systems*. Kluwer Academic Pub.

Harel, D. and Pnueli, A. 1985. "On the development of reactive systems". In *Logics and Models of Concurrent Systems*, Springer-Verlag, Ed., k. r. apt ed., vol. 13 of NATO ASI. New York, pp. 477-498.

Kocik, R. and Sorel, Y. 1998. "A methodology to design and prototype optimized embedded robotic systems". In *2nd IMACS International Multiconference CESA'98* (Hammamet, Tunisia, April).

Landau, I.D. 1988. *Identification et commande des systèmes*. Hermes, 1988.

Le Lann, G. 1990. "Critical issues for the development of distributed real-time computing systems". Research Report 1274, INRIA, (August).

Mathai, J. 1996. *Real-time Systems: Specification, Verification and Analysis*. Prentice Hall.

Nikoukha, R. and Steer, S. 1999. "Scicos : A hybrid system formalism". In *I1 th European Simulation Symposium* (Erlangen, Germany, october).

Ogata, K. 1970. *Modern Control Engineering.* Prentice-Hall International Editions.

Ogata, K. 1987. *Discrete-Time Control Systems*. Prentice-Hall International Editions.

Phillips, C.L. and Troy, N.H. 1984. *Digital Control System : Analysis and Design*. Prentice-Hall International Editions.

Sorel, Y. 1994. "Massively parallel computing systems with real time constraints, the algorithm architecture adequation". *In Methodology Proc. of Massively Parallel Computing Systems Conference* (Italy).

Stankovic, J. A. and Ramamritham, K. 1993. *Advances in Real-Time Systems.* IEEE computer Society Press.

Timmerman, M. 1999. "Rtos market survey preliminary results". *Real Time magazine* (march).

Torngren, M. 1990. "Fundamentals of implementing real-time control applications in distributed computer systems". In *Real-Tune Systems*. Kluwer Academic, pp. 219-250.

QUEUEING SYSTEMS SIMULATION

COMBINATION AND INTERWORKING OF TRAFFIC-FLOW ANALYSIS AND EVENT-DRIVEN DISCRETE EVENT SIMULATION

Gábor Lencse
Department of Telecommunications
Széchenyi István University
Egyetem tér 1.
H-9026 Győr, Hungary
e-mail: lencse@sze.hu

KEYWORDS

traffic-flow analysis, discrete event simulation, communication networks, performance analysis, traffic modelling

ABSTRACT

The traffic-flow analysis (TFA) is a novel method for quick performance estimation of communication systems. It gives approximate results and is potentially faster than event-driven discrete event simulation. The combination of the two methods seems to be a promising approach for the analysis of a system together with its environment. The combined method can be applied especially well for the performance analysis of the critical parts of communication networks.

INTRODUCTION

The traffic-flow analysis (TFA) (Lencse, 2001) is a simulation-like method for rapid investigation of the traffic conditions of communication networks. Unlike discrete event simulation, TFA does not model the travelling of each packet through the network individually, rather it uses statistics to model the networking load of applications. The method distributes the traffic (the statistics) in the network first, and calculates the specific traffic conditions for each line and switching node in the second step. The results are approximate but may characterize the traffic conditions of the network satisfactorily.

The event-driven discrete event simulation (DES) can be used for detailed and accurate analysis of a system, however if the system is large and complex, the number of the events may be so high that the execution requires unsuitably long time even using a supercomputer. Because of the algorithm of the event-driven DES, parallelisation is not an easy task and the achievable speed-up is often limited. (Fujimoto 1990)

This paper deals with the combination of the two methods. Several times we are faced with the following problem: there is a communication network that has a critical part. E.g. we have an X.25 network servicing ATM and POS terminals, and we would like to check what happens if an important link to the server fails. The common characteristic feature of these types of problems is that there is a critical part of the network (like the immediate neighbourhood of the server in the example above) that should be modelled accurately and there is the all remainder part of the network that cannot be omitted because it gives the load for the critical part. Using event-driven DES for the analysis of such kind of networks, we face with the following contradiction: even though we are interested in the critical part, the vast majority of the events in our simulation occur in the rest of the network (because that contains all but some of the nodes, lines, traffic generators). The basic idea of the suggested solution is the following: DES should be used for the precise analysis of the critical part only, and TFA is to be applied for the rest of the network. This solution has the following justification: the critical part is modelled accurately enough, but the computing power is not wasted for the execution of events that are irrelevant for us individually, only their certain statistical consequences influence the behaviour of the critical part of the network.

For the combination of the two methods, the methods should be taught to "speak the same language" to be able to interchange information with each other. It means that we need conversion methods between the *statistics* of TFA and the *events* of DES in both directions. Additionally, we are faced with the following problem: on one hand virtual time (model time) is very important in the event-driven DES, on the other hand TFA does not deal with the elapsing of time, TFA can be used for taking a snapshot of the system in a given state. Our implementation of TFA – a part of a network expert system called Iminet (Elassys, 2004) – was built on top of an event-driven discrete event simulation engine, and uses the simulator's virtual time for the internal purposes of TFA (for determining the spatial distribution of the traffic – will be explained later).

Let us discuss the issue of the conversion between the statistics and messages first, and solve the virtual time problem after that.

BIDIRECTIONAL CONVERSION BETWEEN STATISTICS AND MESSAGES

General Approach

Let us suppose that our network to be examined can be divided into two parts: one of them is to be simulated by DES and the other is to be analyzed by TFA. A system

built up by several DES and TFA segments can be traced back to this simple case.

The basic idea of the conversion between statistics and messages on the boundary of the two segments is shown in Figure 1. When the traffic information is travelling from the DES segment to the TFA segment, the representation mode is changed from messages to statistics. It means that we need to collect the appropriate statistical characteristics of the message flow, and perhaps it is necessary to transform the results to the kind of statistics that are used in the TFA segment. On the opposite direction, where the traffic travels from the TFA segment to the DES segment, messages should be generated on the basis of the statistics (called *traffic model* in TFA terminology) of the TFA segment. The characteristics of the message flow such as the distribution of the message length, inter-arrival time, the source and destination of the packets are representing the same information that is coded in the statistics travelling from the TFA segment to the DES segment.

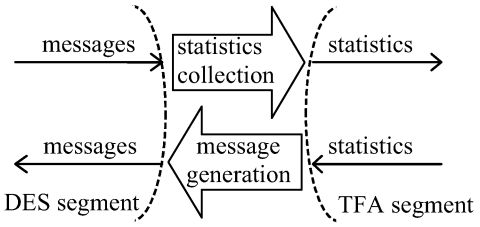


Figure 1. The Basic Idea of Connecting a DES and a TFA Segment

How can these conversions be done between the two representation forms of the traffic? Not too much can be said about it in general, as TFA is a general method which can work with various traffic models that comply with the *requirements for the traffic model* presented in the original paper. (Lencse, 2001) First, we shall examine the problems that the conversion between the traffic description methods of TFA and DES may cause in general, and then we deal with some of the issues in more specific cases.

When doing system modelling, we focus on some features of the system that are important for us and neglect some other details that we consider less important from our point of view. Let us consider the following example: we would like to examine the performance of a network and find its possible bottlenecks. To make a traffic model for DES, we probably carefully examine (for all traffic sources) the distribution of the packet length and of the inter-arrival time as well as the correlation of the two, (and perhaps some other features such us the burstiness of the traffic) but we are not really interested in the contents of the packets (the information they carry). To make a traffic model for TFA, the traffic sources could be characterized by their *packet throughput distribution* (a histogram of the number of packets per seconds) to model the load for the switching nodes and by their *bit throughput distribution* (a histogram of the number of bits per seconds) to model the load for the lines. Using either DES or TFA, we design a model, so that the results of the experiments with the model answer our questions. However, during modelling, a given portion of our efforts is done so that the model meet the requirements of the given simulation system. Such as we need to use the given topology description language, we have to code the activities of the elements in e.g. in C++, etc. It is also true in the case if we would like to combine TFA and DES. ***We must build both models so that they could be combined.*** As for the traffic exchange, it means that the traffic model of both the TFA segment and the DES segment need to contain enough information so that it could be converted to the other type and the result of the conversion will satisfy the requirements of the traffic model of the target segment. This is what we can say in general.

Requirements of a Typical Traffic Model

In the example above, we mentioned bit-throughput and packet-throughput as possible TFA traffic models. They were also proposed and described in details in the original paper on TFA (Lencse, 2001). As they are nothing else than *histograms*, if we choose them, we can use a lot of previous results. A good book in the topic of non-parametric density estimation is (Devroye and Győrfi 1985). The usage of histograms for statistics collection was examined in (Lencse 1998b). That research was done in the interest of the Statistical Synchronisation Method (SSM) (Pongor 1992), which is a less well known, but promising synchronisation method for parallel DES. We can borrow a lot of results of SSM research. A short summary of the *Statistical Synchronisation Method* is given here. See (Pongor 1992) for more information.

Similarly to other parallel discrete event simulation methods, the model to be simulated – which is more or less a precise representation of a real system – is divided into segments, where the segments usually describe the behaviour of functional units of the real system. The communication of the segments can be represented by sending and receiving various messages. For SSM, each segment is equipped with one or more input and output interfaces. The messages generated in a given segment and to be processed in a different segment are not transmitted there but the *output interfaces* (OIF) collect statistical data of them. The *input interfaces* (IIF) generate messages for the segments according to the statistical characteristics of the messages collected by the proper output interfaces.

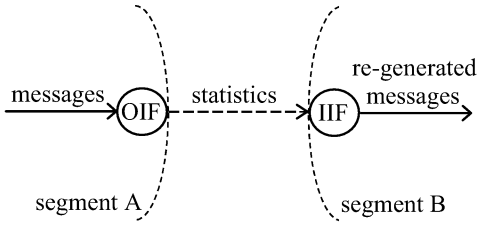


Figure 2. An OIF - IIF Pair

The segments with their input and output interfaces can be simulated separately on separate processors, giving statistically correct results. Events in one segment have not the same effect in other segments as in the original model, so

the results collected during SSM are not exact. The precision depends on the partitioning of the model, on the accuracy of statistics collection and regeneration, and on the frequency of the statistics exchange among the processors.

Histograms were used for representing statistics in the experiments testing the behaviour (Lencse 1997) and the achievable speedup (Lencse 1998a) of SSM.

Despite of the fundamental difference between the parallel DES using SSM and the combination of DES and TFA, they both can use the same or similar statistics collection and message generation mechanisms.

If we use histograms for the implementation of the traffic model of TFA, we can reuse the results achieved in (Lencse 1998b). In that paper, not only histograms, but a number of other statistics collection methods were compared and applied in the case of different distributions in order to determine what non-parametric density estimation methods should be used for statistics collection. The examined methods were: Barron estimate, equidistant histogram (histogram with equal bin/cell width) equi-probable bin histogram and also the relative frequency method for discrete distributions. Both their resource requirements and accuracy were examined. The L_1 error criterion was used for measuring the error of the estimation methods. The conclusions were following. Though theoretically, the equi-probable bin histogram should produce less L_1 error than the equidistant one, and it was also the experience in the case when too few bins were used, but if there were enough bins, the equidistant histogram was better. For continuous distributions, the equidistant histogram is the method of choice. The relative frequency method produced acceptable results for the examined real life discrete or quantized distributions.

It is very likely that histograms will satisfy our needs, and knowing the modelled system well enough we can determine the range and the cell size of the histogram if we are faced to one-dimensional distributions. However it might happen that in some cases we do not know enough a-priori information about some of the distributions or the number of observations is not enough to guarantee the accuracy of the results using histograms. In this case the messages-to-statistics conversion may be done in two steps: First, another method can be used for statistics collection and the results should be transformed to histograms after completing the statistics collection. A good candidate is the K-split method (Varga 1998). K-split maintains the number of cells (i.e. bins) optimal for the distribution and the number of observations by doing cell splits. If we have multi-dimensional distributions, this two step procedure (using K-split first and convert to histograms later) is strongly advised.

DEALING WITH THE DIFFERENT USAGE OF VIRTUAL TIME

The issues of Combination and Interworking

Under *combination* of TFA and DES we mean that we would like to use them for the same network, one method for one part and the other method for the other part of the system. The bidirectional conversion between the two kinds of traffic representations (discussed in the previous chapter) is a necessary precondition; otherwise the two parts can not exchange traffic information with each other. However if we would like TFA and DES to *interwork*, that is actively cooperate with one another, we need more. Before going into the details, let us consider how they use the virtual time – that will be the key issue.

The algorithm of the event-driven DES is the following (FES means the Future Event Set):

```
initialize, insert certain events into the FES;
repeat
  remove the first event from the FES;
  NOW:=the virtual time of the event removed from the FES;
  process the event, during this insert some event(s) into the
        FES if necessary;
until (FES is empty) or (NOW>limit) or (for other reason, we
        must stop)
```

Virtual time plays a key role in this algorithm.

TFA gives a snapshot of the traffic conditions of the examined system, that is, it analyses the network at a given point of virtual time. Though virtual time does not take part in TFA as in DES (as it would increase/elapse during the analysis) it is usually important and influences the results through the parameters of the system. For example, if we model POS (Point of Sale) or ATM (Automatic Teller Machine) terminals in a network, the frequency of the transactions highly depends on the hour of day (and also on the day of week).

Now, let us return to the definition of ***interworking of TFA and DES***. Under it we mean the following cooperation (one or more times it happens that):

1. DES sets up the virtual time of TFA
2. DES provides input parameters to TFA
3. DES calls TFA for the evaluation of the part of the system that was trusted for TFA
4. TFA runs
5. TFA returns parameters to DES

The way these steps are done highly depends of the architecture and communication scheme of the program(s). Let us see some possibilities:

A) The DES part and the TFA part are two processes and communicating with each other by inter-process communications (using PVM/MPI or named pipes, etc.).

B) TFA is a function/procedure within the DES program and is called sometimes, but it does not use the virtual time of the DES engine for its internal purposes.

C) TFA is a set of functions within the DES program and is called sometimes, and it uses the virtual time of the DES engine (and some of its services) for its internal purposes.

All of them could be useful, but may have some weaknesses too. Solutions "A" may not be feasible with commercial DES programs like OPNET Modeler (OPNET, 2004) that are usually not distributed with the full source code and therefore it would be inconvenient or sometimes practically impossible to use inter-process communications.

"B" can be the method of choice, if we use a commercial simulator that is not flexible enough for "C" or for some reason we prefer this solution. By choosing "B" we avoid all the issues to be discussed in the remainder of this paper, but also miss all the benefits of using the functionalities of the simulator. Then we are on our own and have to implement everything ourselves in the TFA part, including probably the topology description and routing of the network. We have no help from the simulator, no or little possibilities for reuse of the ready made elements. If we write a function that is once called, then works and finally returns we cannot use the event mechanism of the simulator, that is: we lose the chance to use all the models that are built up in the event-driven manner.

It is worth choosing "C", if we are able to use a lot a functionalities of the simulator in the TFA part and it will reduce our programming (also model building) efforts significantly. From now on, we concentrate on the different issues we are faced to when choosing "C".

What do the steps of interworking mean if we use the virtual time of the simulation system within TFA? (That is, we use the event mechanism of the simulator.) Let us consider them one by one.

1. DES sets up the virtual time of TFA by scheduling its starting event to the appropriate virtual time

2. DES provides input parameters to TFA by sending messages to the module(s) responsible for TFA

3. DES not at all calls TFA, it runs when its virtual time comes

4. when TFA runs, it may use the event scheduling system, but it needs some discussion (because of the contradictory usage of virtual time)

5. TFA returns parameters to DES by sending messages to the appropriate module(s) in the DES segment

All but no. 4 are quite clear. The issue of virtual time usage is discussed in the text subsection.

How to conciliate the virtual time usage in DES and TFA?

First, let us recall how TFA distributes the traffic in the network. It pursues the following procedure for all traffic sources: first, it divides the statistics (describing the traffic of the given source) into smaller ones, and then sends them to their target one by one. Their size is called the *size of routing unit* (S_{RU}) and they are routed one by one in the network according to the same algorithm that the nodes use for routing the packets in the network (in DES). Of course, S_{RU} have to be chosen so that the granularity should be smooth enough (to produce good enough results) on the one hand; however, the number of routing units should be at least one order of magnitude less then the number of packets would be in DES (to be faster then DES) on the other hand. As we mentioned, the statistics can be sent by the source to the destination as messages, and the switching nodes and lines can forward them as messages. The nodes can use the same routing algorithm for routing the statistics as DES would use for routing the packets. This is an important benefit that we do not need to write everything from scratch. In an object oriented environment we can reuse the granted elements even if we must slightly modify their behaviour. A very important question is that how the statistics that are coming from different sources and travelling through the same network can be interwoven well? This is crucial for the appropriate spatial distribution of the traffic in the network. There are two solutions that both reconcile the virtual time usage of TFA and DES, but in a different manner. With a little bit of irony, we can call the first one *model hacking* and the second one *kernel hacking*.

Under *model hacking* we mean that we have to construct the DES model so that it allow a given small but positive T virtual time for TFA to work. T should be chosen small enough so that the DES model can tolerate that even though it starts TFA at t virtual time, it can receive back the results from TFA at $t+T$ virtual time. For the interweaving of the traffic from different sources, let us consider two sources **A** and **B** that send N and M number of statistics respectively. The virtual times of the sending of the i-th statistics by **A** and **B** are:

$$t_{A,i} = t + i\frac{T}{N}, \qquad i = 0,\ 1,\ \ldots\ N-1$$

$$t_{B,i} = t + i\frac{T}{M}, \qquad i = 0,\ 1,\ \ldots\ M-1$$

When distributing the statistics, TFA does not model processing or transmission delay, so the statistics travel with the same time stamp through the network. A finishing event is scheduled for all the nodes and lines of the TFA model with the time stamp of $t+T$. This event causes that the nodes and lines transform the previously summed up statistics so that their finite capacities are considered. See the

algorithm in the original paper (Lencse, 2001), and the proof of its convergence in (Lencse and Muka, 2005).

The advantage of model hacking is that (unlike kernel hacking) this solution does not require any modifications of the simulation kernel. Hopefully our DES model will tolerate the small T long latency of the TFA segment. Otherwise we need to make some modifications on our DES model – remember, the method is called model hacking.

Kernel hacking provides a cleaner way for the interweaving of the traffic from different sources. However it requires some modifications in the DES simulation kernel. It can be done if either the kernel is our own property just like Elassys Consulting Ltd. owns the Iminet Network Expert System (Elassys, 2004) that contains a DES kernel or the simulator is open source such as the OMNeT++ Discrete Event Simulation System (Varga, 2004). The modification of the simulation kernel means the introduction of sub-time, that is, a second time stamp is added to all the events, and the events are ordered primarily on the basis of the original time stamp, but if the original time stamp of two messages are equal their order is determined on the basis of the second time stamp. Of course, the second time stamp of the normal DES events should be set to zero. Denoting the complete time stamp of the events by separating the original and the second time stamp with a comma, the scheduling times of the TFA statistics sending events in the previous example look like:

$$t_{A,i} = t, i\frac{1}{N}, \quad i = 0, \quad 1, \quad \ldots \quad N-1$$

$$t_{B,i} = t, i\frac{1}{M}, \quad i = 0, \quad 1, \quad \ldots \quad M-1$$

Of course, any other fixed constants could stand in the nominator instead of "1".

CONCLUSIONS

We have examined how the two methods, TFA and DES can be combined for more efficient performance estimation of communication networks.

We have shown how TFA and DES can interchange information using a bidirectional conversion between their traffic representation forms: statistics and messages. We found that our previous research results on statistics collection methods are reusable.

We have shown two ways for resolving the issue of different (and contradicting) virtual time usage of TFA and DES.

We conclude that the integration of TFA and DES is promising and building the TFA package on top of a flexible DES kernel is a practical way of implementation.

ACKNOWLEDGEMENT

This research was financed by the Hungarian Academy of Sciences under the János Bolyai research grant.

REFERENCES

Devroye, L.; L. Győrfi. 1985. *Nonparametric Density Estimation: The L1 view.* John Wiley, New York

Elassys Consulting Ltd. 2004. "Iminet Network Expert System" http://www.elassys.hu

Fujimoto, R. M. 1990. Parallel Discrete Event Simulation. *Communications of the ACM* **33** no. 10, 31-53

Lencse, G. 1997. "Efficient Simulation of Large Systems - Transient Behaviour and Accuracy" *Proceedings of the 1997 European Simulation Symposium (ESS'97)* (Passau, Germany, Oct. 19-23). SCS Europe, 660-665.

Lencse, G. 1998a. "Efficient Parallel Simulation with the Statistical Synchronization Method" *Proceedings of the Communication Networks and Distributed Systems Modeling and Simulation (CNDS'98)* (San Diego, CA. Jan. 11-14). SCS International, 3-8.

Lencse, G. 1998b. "Statistics Collection for the Statistical Synchronisation Method" *Proceedings of the 1998 European Simulation Symposium (ESS'98)* (Nottingham, UK. Oct. 26-28). SCS Europe, 46-51.

Lencse, G. 2001. "Traffic-Flow Analysis for Fast Performance Estimation of Communication Systems" *Journal of Computing and Information Technology* **9**, No. 1, 15-27

Lencse, G; L. Muka 2005. (expected) "Convergence of the Key Algorithm of Traffic-Flow Analysis" *Journal of Computing and Information Technology,* - under review,

OPNET 2004. "OPNET Modeler" http://www.opnet.com

Pongor, Gy. 1992. "Statistical Synchronization: a Different Approach of Parallel Discrete Event Simulation". *Proceedings of the 1992 European Simulation Symposium (ESS'92)* (Nov. 5-8, 1992, The Blockhaus, Dresden, Germany.) SCS Europe, 125-129.

Varga, A. 1998. "K-split – On-Line Density Estimation for Simulation Result Collection". *Proceedings of the 1998 European Simulation Symposium (ESS'98)* (Nottingham, UK. Oct. 26-28). SCS Europe, 41-45.

Varga, A. 2004. "OMNeT++ Discrete Event Simulation System" http://www.omnetpp.org

BIOGRAPHY

Gábor Lencse received his M.Sc. in electrical engineering and computer systems from the Technical University of Budapest in 1994 and his Ph.D. in 2000. The area of his research is (parallel) discrete event simulation methodology. He is interested in the acceleration of the simulation of communication systems. Since 1997, he works for the Széchenyi István University in Győr. He teaches computer networks and networking protocols. Now, he is an associate professor. He does research and development in the field of the simulation of communication systems for the Elassys Consulting Ltd since 1998.

He is a member of IEEE Hungary Section, IEEE Computer Society and IEEE Communications Society.

REFLECTIVE SIMULATION MODELS FOR OPTIMIZATION OF FMS

Eugene Kindler
Ivan Křivý
Ostrava University Faculty of Science
CZ – 701 03 Ostrava, Dvorakova 7, Czech Republic
E-mail: kindler@ksi.mff.cuni.cz, ivan.krivy@osu.cz

Philippe Lacomme
Alain Tanguy
LIMOS, CNRS UMR 6158, Université Blaise Pascal,
Complexe des Cézeaux, 63177 Aubière Cedex, France
E-mail: {lacomme, tanguy}@isima.fr

KEYWORDS
Discrete Event Simulation, Reflective Simulation, SIMULA, Flexible Manufacturing System, Job Scheduling

ABSTRACT

This paper addresses the job and material device scheduling in flexible manufacturing systems (FMS) using a single device Automated Guided Vehicle Systems (AGVS). The problem is solved using a simulation based approach. This approach takes into account the impact of the dynamic behaviour of the system on the problem solving. For one job input sequence, the scheme we propose, computes the optimal device schedule. All device schedules are investigated thanks to the reflective discrete event simulation, which branches to each device destination station depending on jobs waiting in output buffers. The optimal solution knowledge enables us to evaluate the performance of basic device management rules (FIFO, STT, SPT, MOQS) proving these rules provide solutions not ever close to the optimal one. The modelling framework is benchmarked using difficult instances first introduced by Ulusoy and Bilge in 1993.

PROBLEM STATEMENT

The Manufacturing System

The manufacturing system is assumed to consist of one or more Input/Output (I/O) stations and a set of "flexible manufacturing units". A single device AGVS integrates all manufacturing system components into a computer integrated manufacturing system, which represents the FMS. Loads enter the system through one of the I/O stations and when all the operations are performed they exit through one of the I/O stations and leave the system. The generic term "flexible manufacturing unit" (FMU) designates a more basic unit of FMS (Fig. 1) which is called a flexible manufacturing module in (Kusiak, 1985a, b), a flexible machining cell in (Browne, 1985) and a single flexible machine in (MacCarthy, 1993). There is a pick-up/drop off point associated with the FMU. A load is always delivered and dropped off to the input buffer and picked up from the output buffer of a FMU. If the FMU or the device is occupied the parts wait for the FMU or the device to become free in the input buffer or respectively in the output buffer. Comprehensive FMS descriptions are available in (Tchernev, 97) and (Soukhal, 2001).

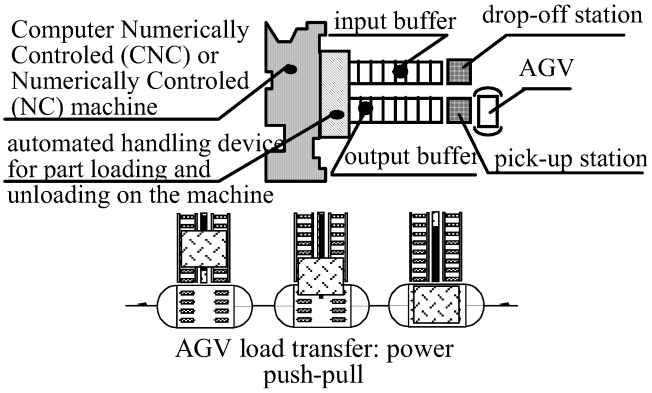


Figure 1: Flexible Manufacturing Unit

Problem Formulation

In a FMS with a finite buffer capacity and a single device AGVS, the job input scheduling and device dispatching can be stated as follows: Given a FMS such as described above, a job input sequence and dispatching rules determine the AGVS move scheduling minimizing the makespan. This problem can be represented as in figure 2 along with the deadlock example. The maximum number of jobs in the FMS is limited. Each machine has an input buffer and an output buffer with capacity equal to 2. The machines *A* and *B* are busy and their buffers are full. Job 1 needs to be carried to busy machine *B* and job 6 must be processed by busy machine *A*. The single AGVS is not able to do its task and the deadlock occurs despite of free buffers of the idle machines *C* and *D*. The FMS deadlock depends on the maximum number of jobs simultaneously processed in the FMS, the job input sequence, the job schedule and the AVGS control.

Since the number of jobs simultaneously allowed in the system is limited and the total processing time is constant (predetermined upon arrival of jobs), the only way to minimize the makespan consists in reducing the waiting times due to the blocking and non availability of required resources. Therefore, the aim is to find a job scheduling that satisfies the following set of objectives in the best way:

- to reduce the cumulative lead time of the production schedule;
- to improve the machine utilisation;
- to reduce the total loaded and empty travels of the AGV;
- to guarantee the system operates without deadlock.

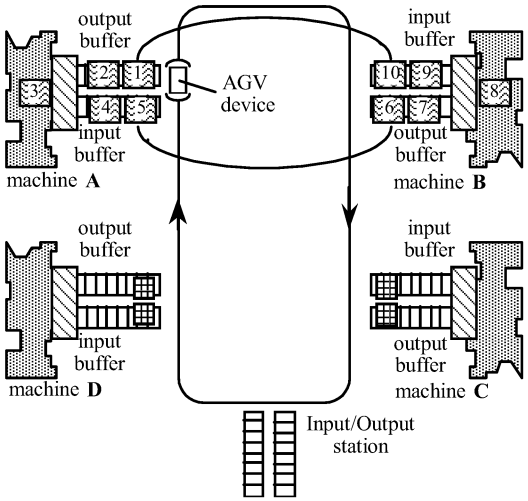


Figure 2: Problem Constraints and Deadlock in FMS

The deadlock problem was studied in (Kindler *et al.* 2003) and a solution was proposed in a deterministic context. It is based on a nested simulation model (reflective simulation) that forbids job entries and anticipates the FMS behaviour until all the jobs are processed and carried to the exit or a condition insures the completion of the treatments. When a deadlock is detected, the current job entry is delayed until the deadlock disappears. This kind of simulation models is efficient and very promising. So, it was foreseen to design multiple nested simulation models in order to find the optimal moves of the guided vehicle. A job entry has ever the highest priority but when the maximum number of jobs is reached or many jobs are waiting in front of output buffers, one move must be selected from the set of possible ones. It is generally defined by one single management rule, for instance the nearest waiting job is chosen. We propose to anticipate the behaviour of the FMS when more than one decision may apply. Of course, a new decision depends on previous ones, therefore the main FMS simulation model has to start a new FMS simulation model when a decision must be tested, and so on. That implies to handle many nested simulation levels of the system.

REFLECTIVE SIMULATION

We introduce the principles of the reflective simulation defined as a system simulation that simulates the system or some of its parts by the way of nested simulations.

Motivation and Problems

The classic simulation is commonly applied during the design of modern systems. Some of them include one or more computers and thus the ability to simulate their future behaviour. Other systems include intelligent components, such as drivers, who are able to determine their shortest path among obstacles and to elaborate decisions. Everybody knows that a simulation run is time consuming and other methods are better but they demand more time for their implementation, their tuning and their adaptation when the system in design evolves. Beside its worst feature, the simulation is often the most suitable method to take into account all the managing constraints of the system, new management rules or configurations and to validate decisions applied during the system functioning. Thus, we have to regard the simulation of simulating components and to study the nesting of an internal model into an external one.

At the beginning of the third millennium, we thought that two model levels could be sufficient. One level is not enough because a simulation backtracking is very difficult and it does not describe the simulated system inside simulating elements. The need of at least two Newtonian time axis, one for each model, appeared as a problem because common simulation tools are not designed to manage more than one time axis. Among other problems, the entity handling of both models may be corrupted by pointer management errors. We remarked that it is not necessary to simulate the simulating components (computers, thinking drivers) and we may generally neglect their computation times or handle them as a short delay. We have only to consider the internal model and its management by the external one. So, we developed a software (Křivý *et al.* 2002) for the automatic generation of both models from a suitable specification of a stand-alone model of the system. Presently, the simulation nesting is not limited to two levels because the decisions must be tested by new nested internal models. Previous research introduced a taxonomy and systematics of nested simulation (Kindler *et al.* 2003a, 200b).

Reflective Simulation Implementation

The implementation of reflective simulation taught that it is a difficult activity because of psychological aspects of the modeller and of a risk of mixing elements of both models. Therefore, we had to define a methodology to design reflective simulation models (Kindler *et al.* 2001, 2002, 2003c). It appeared that we need a simulation language being object-oriented, block-oriented and able to manipulate local classes, life rules and quasi-parallel systems (QPS). Only four candidates were found: Beta, Java, MODSIM and SIMULA. Limitations concerning QPS eliminated Beta and those concerning local classes eliminated MODSIM. Java is not secure regarding element mixing of the different models. Thus, only SIMULA is able to implement reflective simulation in an absolutely secure way. Mainly, there are three free SIMULA compilers: CIM, Lund and Simula a.s. compilers covering the user's needs under UNIX, Linux and Windows. Despite of security problems we decided to develop two models one in Delphi and the other one in SIMULA to validation purpose.

REFLECTIVE SIMULATION MODELS

The model analysis and specification is based on a breakdown into Physical, Logical and Decisional subsystems. As introduced in the first section, the system management is based on three management rules:

- the buffer management rules (FIFO in our study);
- the vehicle management rule when jobs require transport;
- the idle vehicle management rule when no transportation is required (no job in the output buffer of the stations).

The rule we use is STOP since it is a widespread rule, thanks to its great interest for saving power of vehicle. The vehicle management rule is composed of two parts. First, as

soon as the number of jobs in the system is less than the maximal number of jobs allowed, the vehicle must serve first the entry station if possible. Second, when one or more than one job are in an output buffer, the vehicle is managed using a priority rule. Common rules:

- *FCFS - First Come First Served* rule. Under this rule, a device is dispatched to the FMU with the earliest job completion date.
- *STT/D - Shortest Travel Time/Distance* rule. This rule aims at minimizing the travel time of empty devices. When the device has just finished a drop-off, it is dispatched to the nearest pick-up station.
- *SPT - Shortest Processing Time* rule. Under this rule, a device is dispatched to the pick-up station having a job with the smallest processing time at the next station.
- *MQS - Maximum Queue Size* rule. Under this rule, a device is dispatched to the station which output buffer has the largest number of jobs. If two or more stations have the same output buffer size then the conflict is solved using the redundant device dispatching rule STT.

Obviously, these rules provide sub-optimal solutions if more than one job require to be carried from its current station to the next one. We propose to replace this second rule by an enumerative method in order to obtain the optimal vehicle sequence of transport moves. We denote *Decisional State* any system state where more than one job require to be transported. For each *Decisional State,* the simulation model is suspended and a new one starts for the current system state with one vehicle destination. All the possible destinations are scanned and a tree of all the sequences of decisions is generated. The process is iterated as long as there are jobs to carry.

Delphi Model

The discrete event simulation model is composed of one scheduler and of an object-oriented model of FMS including one AGV. The Scheduler focuses on event management and updates the entity states including machines, buffers and vehicles (Fig. 3). Thanks to the Delphi language, the decision tree is recursively generated. This scheme is a powerful enumerative method which takes into account all the management constraints of the system (input/output buffer capacity, entry priority…).

	Scheduler
Step 1.	**While** (there is events to treat) **Do**
Step 2.	**For** all idle vehicle **Do**
Step 3.	Vehicle := Supervisor.Select_Device
Step 4.	Dest := Supervisor.Compute_Destination
Step 5.	**EndDo**
Step 6.	Supervisor.Manage_System
Step 7.	**For** all vehicle **do**
Step 8.	**If** (Vehicle.State = Empty) **Then**
Step 9.	**If** (there is no job to treat) **Then**
Step 10.	DM := Date of the next event
Step 11.	Simulation_Time := DM
Step 12.	Vehicle.Activity := Idle
Step 12.	**Endif**
Step 13.	**EndIf**
Step 14.	**EndDo**
Step 13.	Supervisor.Update_System
Step 14.	**EndWhile**
Step 15.	Return the performance criteria

Figure 3: Scheduler Main Steps

SIMULA Model

This object-oriented simulation model is based on a simplified view of the general FMS one. It is hierarchically defined.

- The main programme collects information needed to realize the dynamic allocation of several arrays, the numbers of machines, jobs, the buffer capacity and the size of bills of material.
- The first sub-block initializes the constant information such as the job input sequence and the bills of material. It contains the declaration of the *FMS* class and creates the first instance of the simulation model at the top level (external model at level 1). The class parameters inform the model about its level and its upper model that allows communication between models.
- The *FMS* class contains the local classes of the system entities, mainly *Machine*, *Vehicle* and *Job*. The initialization part and the simulation kernel are defined in the life rules coroutine of *FMS* class.

When a FMS class instance is created, it starts its coroutine that initializes the external model (level 1) or makes a copy of the upper model state attributes into the nested internal model (level > 1). The selected transport move is prepared then the internal simulation starts and runs until several decisions have to be tested. Then each decision is anticipated using a new nested internal model (level+1) which reports its best makespan and terminates. One could modify the model behaviour in order to limit the internal simulation time applying a partial search on the decision tree.

The simulation kernel is event driven and based on the SIMULA *Simset* class that performs an efficient event list management. Three machine states are defined: *Idle*, *in Service* and *Blocked.* Only one machine event needs to be scheduled: the end of service. There are more vehicle states: *Loading*, *Carrying*, *Unloading*, *Empty moving*, *Idle* and *Decision.* The end of *Loading*, *Carrying*, *Unloading*, *moving* and *Decision* are scheduled into the event list. For a single vehicle, the transition *Idle - Decision* insures that all simultaneous events concerning the machines are treated early. So, the input and output buffers are up to date and the vehicle can examine all the possible decisions. This reflective simulation model was quickly and easily implemented in SIMULA.

NUMERICAL EXPERIMENT

In this section, we present numerical examples so as to evaluate the proposed models and to compare the AGV management rules. The experiments were carried out on a 2.66 GHz Pentium IV processor under Windows XP operating system.

Networks and Data

For the procedure evaluation we use the layouts and job set of (Ulusoy 1993). These layouts are based on the basic types presented by (Kusiak 1985b). Each Flexible Manufacturing Units is composed of input and output buffers with a limited capacity. In order to limit the magnitude of this study, we assume there are an equal

number of places at each buffer that may not always hold in practice. The buffer capacity is limited to two places. The Input/Output station is assumed to have a sufficient capacity to store all jobs to be scheduled. The other characteristics of these systems are:

- the pick-up and drop-off operations at stations are assumed to have a 1 minute duration;
- acceleration and deceleration have not been considered;
- any battery recharging is done during the off shift period;
- each load station and unload station can be reached by AGV located at any other station;
- vehicle breakdowns never occur.

The experiment focuses only on job-set 1 and layout 1. The tables 1 and 2 summarize the AGV move durations and the sequence of treatment times on specified machines for each bill of material B1 to B5. The following rules apply for the job selection by device: FCFS, STT/D, SPT and MQS. The following extra notations are introduced:

n number of jobs to schedule
N number of jobs simultaneously allowed
I number of investigated solutions
L depth of the search tree

Table 1: Transportation Times

	L/U	M1	M2	M3	M4
L/U	0	6	8	10	12
M1	12	0	6	8	10
M2	10	6	0	6	8
M3	8	8	6	0	6
M4	6	10	8	6	0

Table 2: Job-set 1, Treatment Times on Machines

B 1	L	1: 8	2: 16	4: 12	U
B 2	L	1: 20	3: 10	2: 18	U
B 3	L	3: 12	4: 8	1: 15	U
B 4	L	4: 14	2: 18	U	
B 5	L	3: 10	1: 15	U	

Strategies

The experiments were carried out with two strategies:

- first, the vehicle is not supposed to be idle when one job requires transportation;
- second, the vehicle may wait when one job requires transportation, as long as another one reaches the empty output buffer of its station.

The first strategy avoids unproductive wait of the vehicle and limit the search tree generated by the reflective simulation. This strategy is denoted RS for **R**eflective **S**imulation. The second one generates unproductive idle time for the vehicle but permits to investigate more solutions for the vehicle management. This strategy is denoted **R**eflective **S**imulation with **E**xtensions.

Basic Priority Rules Performances

Table 3 reports the performances of FIFO, STT, SPT, MOQS and regarding the optimal management of the vehicle – column RS and RSE. One may note that as soon as the number of jobs simultaneously allowed in the system is greater than 2, there is a significant improvement of the solution using RS strategy.

For N=1 all the priority rules propose the same makespan. For n=N=3, the **RS** strategy provides a solution value 194 lower than 198 – STT rule result. The difference increases for n =4 or 5. When a sufficient number of jobs are allowed in the system, the study proves the basic priority rules are strongly sub-optimal.

Table 3: Experiments for the Job-set 1 and Layout 1

n	***N***	**FIFO**	**STT**	**SPT**	**MOQS**	**RS**	**RSE**
2	2	132	132	132	132	132	132
3	2	207	207	207	207	**207**	**192**
3	3	202	**198**	216	**198**	**194**	**188**
4	2	236	236	236	236	236	236
4	3	268	**252**	260	**252**	234	234
4	4	248	252	264	**242**	232	232
5	2	299	299	299	299	**299**	**283**
5	3	308	**290**	294	**290**	**278**	**268**
5	4	303	**284**	308	**284**	268	268
5	5	306	294	322	**284**	**278**	**276**

Strategy Evaluation

The RSE strategy explores an enumerative tree larger than this of RS strategy. However, the results obtained with RSE strategy are often close to the RS results. No improvement is obtained for 2 and 4 jobs to schedule. For 3 jobs to schedule RSE provides 192 face to 207 with RS for 2 jobs simultaneously allowed in the system. This conclusion remains true for 3 jobs simultaneously allowed. RSE provides a significant improvement of the value 194 first computed with RS strategy. We note improvements occur only 5 times regarding the 10 instances we used.

Scheme Performance

The number of solutions investigated by RSE strategy is greater than the number of solutions investigated by RS strategy. Note the number of solutions investigated with RSE strategy is sometimes one hundred times greater. Because for 5 instances, no improvement is obtained with RSE, one may conclude: including vehicle wait in the search process is not a promising approach as regards the number of investigated solutions.

Table 4: Number of Investigated Solutions

n	***N***	**RS**	**I**	**RSE**	**I**
2	2	132	1	132	45
3	2	207	1	192	373
3	3	194	29	188	10,787
4	2	236	1	236	3,537
4	3	234	59	234	75,959
4	4	232	1,241	232	196,292
5	2	299	1	283	25,251
5	3	278	174	268	1,268,347
5	4	268	5,565	268	3,030,266
5	5	278	34,036	276	2,806,952

Large Scale Instance Resolution

RSE strategy is more time consuming and resolution of large scale instances with more than 5 jobs can only be investigated with RS strategy. Table 5 reports results for 6 jobs to schedule. Note the significant number of solutions investigated and the depth level of reflective simulation. For N=6 the search tree is composed of 16 levels.

Table 5: Large Scale Instances Resolution with RS Strategy

n	*N*	FIFO	STT	SPT	MOQS	RS	I	L
6	1	436	436	436	436	436	1	1
6	2	349	349	349	349	349	1	1
6	3	352	336	355	336	324	454	12
6	4	353	322	370	326	310	24,912	16
6	5	368	326	360	312	308	238,574	16
6	6	342	334	382	336	318	1,125,915	16

Reflective Simulation Performances

Very high performances can be reported for both simulation models. The RSE strategy investigates 3,030,266 solutions in 262 s, for 5 jobs and N=4. One solution only requires 86.46 µs average time to be enumerated. Note the number of executions of the simulation model is strongly greater than 3,030,266 including partial solutions under construction. There is no significant computational time difference between Delphi and SIMULA implementation of the models.

Critical Comments

Resolution of large scale instances will require sophisticated branching strategy and powerful lower bound to avoid whole tree enumeration. To obtain a full branch and bound scheme it could be useful to include previous works on lower bound (Lacomme *et al.*, 2002). (Espinouse et al., 2000) proposed:

- a branch and bound strategy to determine the best job sequence;
- a discrete event simulation model with basic priority rules (FIFO, STT…) for vehicle management.

A promising extension of the scheme we propose, consists in including the search of the best job input sequence in the reflective simulation model.

CONCLUSION

A very difficult combinatorial problem is investigated using a reflective simulation based approach. To the best of our knowledge, it is the first attempt to optimally solve the problem of vehicle scheduling with all the managing constraints of flexible manufacturing systems including but not limited to: input/output buffer capacity, idle transportation time and limited number of jobs simultaneously in the system. This work is a step forward exact resolution of jobs sequencing and vehicle scheduling for medium and large scale instances. Two discrete event simulation models have been designed and implemented in SIMULA and Dephi languages. Each model validates the other one. The reflective simulation approach provides high performance models: one solution requires less than 100 µs to be enumerated. Our research is now directed to:

- determination of the optimal job input sequence;
- branching strategy during reflective simulation;
- improved pruning strategy based on lower bound computation to save time during the decision tree enumeration.

REFERENCES

Browne, J., Dubois,D., Rathmill,K., Sethi,S. P., and Stecke, E., 1984, "Classification of Flexible manufacturing systems". *FMS Magazine*, (2) , 114-117.

Espinouse, M-L, P. Lacomme, A. Moukrim and N Tchernev. 2000. "A Branch and bound approach for the job input sequencing and devices dispatching in FMS based on AGVs". Congrès IFAC/IFIP/IEEE Management & Conduite en Production & Logistique (MCPL'2000, 5-8 july), 67-72.

Kindler E.; I. Křivý and A. Tanguy. 2001. "Reflective simulation of discrete logistic and production systems". In: E. H. J. Kerkhoffs and M. Snorek (eds.): Modelling and Simulation, ESM2001, Prague, 6-9 June, 2001. SCI, Delft, 861-865.

Kindler E.; I. Křivý and A. Tanguy. 2002. "Reflective simulation of logistic and production systems" In: A. G. Bruzzone, Z. Merkuryev, R. Mosca (editors): International Workshop on Harbour, Maritime and Multimodal Logistics Modelling & Simulation HMS, MAS 2002, Genoa, 97-102

Kindler E., I. Krivy, A. Tanguy. 2003a. "Systematics of Nested Simulation". In: Y. Merkuryev, A. G. Bruzzone, G. Merkuryeva, L. Novitsky, E. Williams (editors): The International Workshop on Harbour, Maritime and Multimodal Logistics Modelling & Simulation HMS 2003. 18-20 September, Riga, Latvia. 191-195

Kindler E., I. Krivy, A. Tanguy. 2003b. "Simulation Of Simulating Computerized Systems". A. G. Bruzzone, R. Mosca (editors): International Workshop on Applied Simulation, Modelling & Applied Simulation MAS 2003, 2-4 October, Bergeggi, Italy, 16

Kindler E., I. Krivy, P. Lacomme, A. Tanguy. 2003c. "Simulation of FMS Including Automated Guided Vehicle". ESMC Conference Modelling and Simulation, 27-29 October, Naples, Italy, 122-126

Křivý, I.; E. Kindler and A. Tanguy. 2002. "Software for Simulation of Anticipatory Production Systems". International Journal of Computing Anticipatory Systems, Vol. 11, 320-335

Kusiak, A. 1985a. "Material Handling In Flexible Manufacturing Systems" Material Flow, (2), 79-95.

Kusiak, A. 1985b. "Flexible Manufacturing System: a Structural approach" International Journal of Production Research, 23 (6), 1057-1073.

Lacomme, P., A. Moukrim, N. Tchernev. 2002. "A New Lower Bound for scheduling of FMS based on AGV Material Handling", IFAC'2002, (15th Triennial World Congress) Barcelona, 21–26 July.

MacCarthy, B.L. and J. Liu, 1993, "A New Classification Scheme For Flexible Manufacturing Systems". *International Journal of Production Research*, 31 (2), 299-309.

Soukhal A. 2001. "Ordonnancement simultané des moyen de transformation et de transport". Ph. Dissertation. Université de Tours (France).

Tchernev, N., 1997, "Modélisation du processus logistique dans les systèmes flexibles de production". Ph. Dissertation. Université Blaise Pascal (France).

Ulusoy, G. and Bilge, Ü., 1993, "Simultaneous Scheduling of Machines and Material Handling System in an FMS". *International Journal of Production Research*, 31 (12), 2857-2873.

VISUAL DYNAMIC MODELING OF TRANSPORT FLOWS FOR ELECTRONIC BUSINESS PROBLEMS

Anatoly Levchenkov
Transport and Machinery Faculty
Riga Technical University
1, Kalku St.
LV-1058 Riga, Latvia
E-mail: levas@latnet.lv

Andrew Mor-Yaroslavtsev
Riga International School of Economics and Business Administration
1-2, Meza St.
LV-1048 Riga, Latvia
E-mail: andrew_my@riseba.lv

KEYWORDS

Software Agent, Logistics

ABSTRACT

The purpose of this paper is to develop and study algorithms for interaction of Software Agents in group decision making support systems, intended for logistic problems of control of transport flows in electronic business.

The paper regards the problem of modeling and solution of the problem of minimization of transportation costs and idle time of vehicles for the chosen version of the route and schedule at multiple urban crossings.

The authors offer a generalized algorithm for solution of the problem of drawing up an optimum schedule for traffic through multiple urban crossings with the use of Software Agents and computer models of transport flow. It is also supposed that Software Agents can change the duration of one-type operations on separate processors and can interact in search of the best solution.

At the same time modeling program is a Software Agent controlling vehicles under consideration, while the program of data collecting and analysis is a Supra Software Agent, which makes the optimum processors' operating schedule with the help of a group decision making support system. The processor is the traffic lights of the urban crossroads, and the operation is handling of a transport flow to an assigned direction and assigned time.

INTRODUCTION

The successful solving of electronic commerse tasks and goals achievement depends a lot on a right organization and control of urban transport and pedestrians flow, which provide timely supply of goods and services to customers. This task is especially actual for modern large cities, where transport and pedestrians flow dencity is permanently growing with an invariable carrying capacity of the streets.

With the help of the developed algorithm from a set of cars for particular crossroads applying the Monte-Carlo method we select a certain car. Its parameters are specified in the model: number of a flow, number of the car, velocity, acceleration and distance up to the traffic lights, braking distance and the time left for a prohibiting signal of the traffic lights. On the basis of the distance up to the traffic lights and the braking distance a conclusion on possibility to stop before or on a line at the signal of the traffic lights can be made. Optimum data about the processors' operating schedule received from the model are passed on by the Software Agents interacting with the database management system, recorded in previously prepared database table and displayed on the screen for users of a group decision making support system.

MATHEMATICAL STATEMENT OF THE RESEARCH PROBLEM

The following variables will be introduced:

The given flow (set) of *vehicles*

$$H = (h_1, h_2, \ldots, h_i, \ldots, h_k);\ h \in H$$

i — is an index of a car,

The number of traffic lanes in one direction is known

$R = (r_1, r_2, \ldots, r_j, \ldots, r_m)$

Where j — is an index of a flow,

$v_{i(t)}$ — speed of i vehicle at the time moment t

sl— distance of I vehicle till k-th traffic light at the time moment t,

a — acceleration of i vehicle at the time moment t

t — time,

b — braking distance of I vehicle till k-th traffic light at the time moment t,

x, y — position data of i vehicle at time moment t

sp — distance of i vehicle at the time moment t till a car behind,

sa — distance of i vehicle at the time moment t till a car nearby,

sn — distance of i vehicle at the time moment t till a car in front of it,

TLk – duration of operation of L section for κ traffic light

$V_{\max}$ – maximum permissible value of speed on j traffic lane for vehicles

$V_{\min}$ – minimum permissible value of speed on j traffic lane for vehicles

The elements of modern graph theory, maximum flow function, can be used in modelling of vehicles flows.

We will consider the function which is defined for oriented graph $G = (V, E)$ in the following way:

1. each curve $e \in E$ is confered with positive rational quantity $c(e) \geq 0$ — its carrying capacity,
2. there is one top point v_0 — source $\Gamma^{1}_{v0} = \varnothing$,
3. there is one top point v_z — flow $\Gamma_{vz} = \varnothing$.

Then oriented graph G is called a transport network and marked as $G(T)$.

Let us define flow function $\varphi(e)$ in transport network $G(T)$ as a curve function with the following features:

1. $\psi(e) \geq 0$,
2. $\psi(e) \leq c(e)$,
3. for all $e \in E$: $\sum_{e \in E_v^-} \psi(e) = \sum_{e \in E_v^+} \psi(e)$,

   where $v \neq V_0$, $v \neq v_z$, $v \in V$; E^-_v is at the top point v set of input curves and E^+_v is set of output curves.

At each top point v the sum of input flows is equal to the sum of output flows.

We are interested in the maximum value of flow function

$$\max \sum_{e \in E_v^-} \psi(e) = \max \sum_{e \in E_v^+} \psi(e) = \psi_z .$$

We will mark with A transport network's $G(T) = (V, E)$ set of top points $A \subset V$, which contains top point v_z ($v_z \in A$), and does not contain top point v_0 ($v_0 \notin A$). As a section in transport network $G(T)$ we will call curve set E^-_A, which goes to set of top points A. This notion differs from the considered before notion of section in an oriented graph. The section in the previous meaning contains also curves, which go from set of top points A.

The sum of all section curves carrying capacity is called carrying capacity $c(E^-_A)$ of section E^-_A, e.g.,

$$c(E_A^-) = \sum_{e \in E_A^-} c(e) .$$

In the same way we can define also the value of flow $\psi(E^-_A)$ in section E^-_A:

$$\psi(E_A^-) = \sum_{e \in E_A^-} \psi(e) .$$

As flow φ in each curve e does not exceed its carrying capacity $\psi(e) \leq c(e)$, then there the following dependence is valid

$$\psi(E_A^-) \leq c(E_A^-) .$$

The let us consider Ford (L. R.)–Fulkerson (D. R.) method for maximum flow defining $G(T)$. It was elaborated in the middle of 50s. The method consists of two parts: in the first part we will define the top points, in the second one we will increase flow function ψ, if top point v_z is defined. If top point v_z has not got a definition, then maximum flow ψ_z in the given transport network $G(T)$ is obtained.

The top point definition is a pair, where the first element is top point, from which the first element gets its definition, and the second one indicates its increasing (at the beginning it is 0).

The defining of top points is made in two ways. The top point v_0 is defined $(-, \varepsilon(v_0) = \infty)$, which is not changed. We will choose top point v_i defined as $(v, \varepsilon(v_i))$, then all top points $v_{j1}, v_{j2}, \ldots, v_{je}$, which correspond to the restrictions

$$\psi(v_{jk}) = \min [\varepsilon(v_i); c(v_i, v_{jk}) - \psi(v_i, v_{jk})].$$

The second way to define top points $v_{j1}, v_{j2}, \ldots, v_{jl}$, which correspond to the restrictions $\psi(v_{jk}, v_i) > 0$, is the following: choose designation

$$(v^-_i, \varepsilon(v_{jk})),\ k = 1, 2, \ldots, l,$$

where

$$\varepsilon(v_{jk}) = \min [\varepsilon(v_i);\ \psi(v_{jk}, v_i)].$$

If top point v_z is defined as $(v^+_{i1}, \varepsilon(v_z))$ and flow $\psi(v_{i1}, v_z)$, then curve of a new flow (v_{i1}, v_z) is $\psi(v_{i1}, v_z) + \varepsilon(v_z)$ and transition to other curve flows is changed.

If top point v_{i1} is defined as $(v^+_{i2}, \varepsilon(v_{i1}))$, then curve (v_{i2}, v_{i1}) with flow $\psi(v_{i2}, v_{i1})$ is increased for $\varepsilon(v_z)$ and new flow is

$$\psi(v_{i2}, v_{i1}) + \varepsilon(v_z).$$

If top point v_{i1} is defined as $(v^-_{i2}, \varepsilon(v_{i1}))$, then curve (v_{i2}, v_{i1}) flow is decreased for $\varepsilon(v_z)$ and new flow is

$$\psi(v_{i2}, v_{i1}) - \varepsilon(v_z).$$

It is continued while top point v_0 is reached, then flow function $\psi\varphi$ is increased for value $\varepsilon(v_z)$. The definings of all top points are canceled except v_0, and top point are started to be confered with new definitions. If we can not confer top point v_z with a definition then flow function can not be increased.

The first value $v^{\pm}$ of definition pair shows the shorten curve way from top point v_z to v_0. The second value ε shows possible flow increasing. Let us start with flow $\psi = 0$ for all $G(T)$ curves.

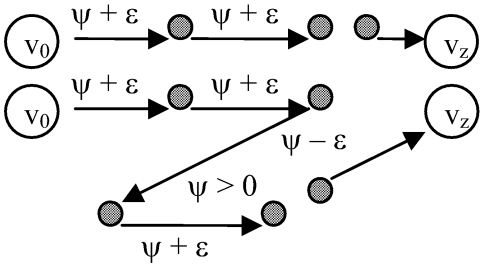


Fig.4. Both cases of flow increasing

The considered graph theory method is applied in this article to solve the task of transport flow modelling.

It is considering the task of such t, lk, V_{max}, V_{min} defining, when H is maximum. The following algorithm is offered to solve this task.

ALGORITHM OF PROBLEM SOLUTION

Step 1. Traffic Department Manager on the client-side (K) enters an inquiry for solution of this complicated problem, using interface (Wr) and information function f Kr:

Step 2. Supra program agent divides this problem into several subproblems

Step 3. To solve multi-criteria assignment problem, when the client has to choose the roads, and the criteria are both quality and quantity criteria, using the server (fig.1.).

Step 4. Supra agent starts inquiry processing with interface (Wr) using information function (procedure) ƒKr: If it receives the answer from server it can move to the next step. If the answer is not received, it has to search it in other server; otherwise the solution does not exist.

Step 5. Server (Wd) starts receiving an answer from data base (Dp) using function (procedure) ƒdp.

Step 6. Server (Wd) starts receiving client's task solution (Pd) using function ƒpd.

Step 7. Server (Wd) starts sending client's task solution to the server (Wr) using function ƒdr.

Step 8. Server (Wr) starts sending client's task solution to Supra program agent SPa that makes solution (s) analysis.

Step 9. To make up a schedule, Algorithm Steps analogue 3-8 are carried out.

Step 10. To analyze and evaluate made up traffic lights' work schedule.

Step 11. Server (Wr) starts sending client's task solution to Supra program agent SPa that makes solution (s) analysis to offer the client only the best task solution using function ƒrSPa.

Supra agent (SPa) offers to the client (K) the best solution using function ƒSPaK.

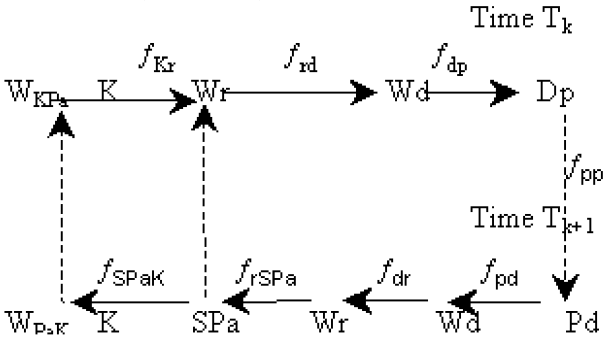


Figure 1: fragment of task solution algorithm.

ROLE OF SUPRA AGENT

After solving one logistics task it is necessary to solve a new one. Supra agent coordinates the task of program agent that has to find an optimal solution for all sub problems.

SUPRA AGENT ALGORITHM FOR FIXING ON DATA ABOUT FLOW IN DATABASE

Step 1. Supra agent connects to data base of crossroad Dp.

Step 2. All variables of the flow for i = 1 till N about vehicles are confered with the current meanings.

Step 3. Preparation of SQL query.

Step 4. Fulfillment of the enquiry, writing into data base for the further calculations.

RESULTS OF EXPERIMENTAL APPROBATION OF THE MODEL

This part considers the results of the experimental approbation of the model. Figures 4 and 5 demonstrate different moments of transport flow in the zones of crossroads. The main goal of the experiment was to check whether it is possible for vehicles data (speed, acceleration, braking distance and time and possibility to stop it in front of the crossroad) to be put into a specially developed data base.

Later on, the fixed data can be applied in the mentioned above mathematic models of transport flows as well as for evaluation of different variants of possible urban transport flow organization, for example public transport zone, arrangement of new traffic lights and highway striping and consequence of these measures.

The process of such a crossroad visual modeling could be considered in the Internet address: http://ecomm.rsebaa.lv/studenti/mja/

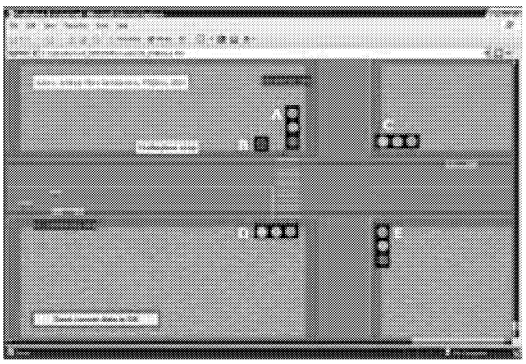


Figure 5. There are additional participants of the flow: two cars, which are moving with the speed higher than that of the trolley.

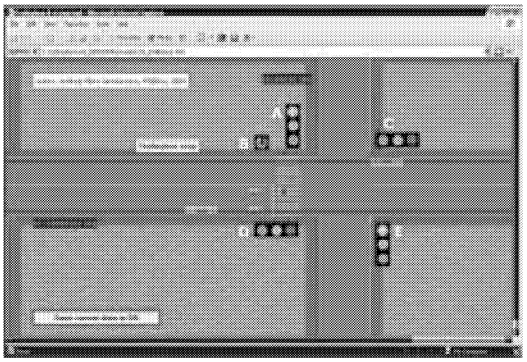


Figure 6. There is a red signal of traffic light for the transport in Kr. Valdemāra street and green one for pedestrians. Vehicles are waiting while the latter cross the road.

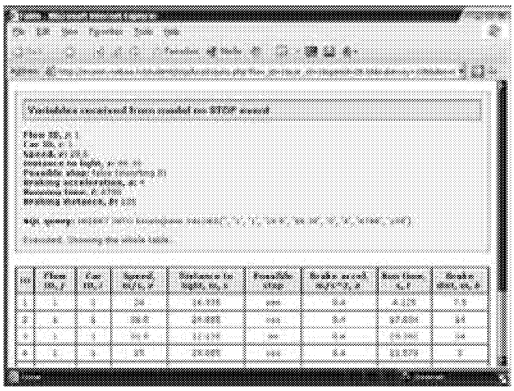


Fig.7. Fragment of a specially developed data base, which keeps necessary data on vehicles participating in the flow. These data are input with pushing the button "Save current data in DB", which is shown in the previous figures.

CONCLUSION

The problem of transport flow in modern cities under the condition of hard crossroads is becoming more important taking into account the increasing density transport flow and limited carrying capacity of the streets.

As this work demonstrates the mathematical and real transport flow modeling results do not correspond enough to the real situation because of quite many assumptions and restrictions, thus the transport flow control finds its application quite seldom.

Some progress in this field could be achieved applying the program of visual dynamic modeling of transport flow developed in this work. The program can be input with parameters of transport flows and crossroads characteristics, which allow to bring the developed model near the real situation of a hard crossroad zone.

One of the possible variants to apply data base is using of the algorithm published before [4].

The work demonstrates the principal possibility for visual modeling of transport and pedestrians flow at crossroads and urban transport stops, that proves the validity of the model.

We assume that after some improvement it could be successfully applied for automatization of urban transport or in automatically regulated systems with a correspondent material and technical basis and crossroads accessories.

REFERENCES

1. Ashton W. D. The Theory of Road Traffic Flow. Methuen; London, 1966.
2. Jonson J., Wood D., Wardlow D., Murphy P. Contemporary Logistics. Prentice Hall. NJ, 2002.
3. Russell S., Norvig P. Artificial Intelligence:a modern approach. Prentice-Hall Canada, Inc., Toronto, 1995.
4. Levchenkov A., Smyshlaev I. Algorithm for program agents in e-learning for solving logistics supply chains problems, ICS-2003 Proceedings, Valencia, 2003.

ACKNOWLEDGEMENT

The authors of this article gratefully acknowledge financial and moral support provided by Latvian Academy of Science, Riga Technical University and Riga International School of Economics and Business Administration.

The authors are grateful to "RICA Digital" student Alexander Dolganov for technical consultation.

BIOGRAPHY

ANATOLY LEVCHENKOV is Associate Professor of the Logistics and Telematics at Riga Technical University from 2001. Since 1997 he has been Associate Professor at the Transport Railways Institute and from 1992 to 1997 he was the Head of the Riga Technical University's Department of Decision Support System. He studied Electrical Engineering in Automation and Telemechanics from 1964 to 1969 and from 1969 to 1992 was a reader and Associate Professor at the Department of Automized Control Systems at Riga Technical University.

ANDREW MOR-YAROSLAVTSEV at present studies at Riga International School of Economics and Business Administration, Riga, Latvia. Simultaneously he studies computer science at Educational and Consulting Centre "RICA Digital", Riga. Participated in Annual Student Scientific Conference in Riga Technical University in 2004.

On Measuring the Supply Chain Flexibility

J. Manuel Feliz Teixeira
António E. S. Carvalho Brito

February 2004

GEIN – Section of Management and Industrial Engineering
Faculty of Engineering of the University of Porto, PORTUGAL
Emails: feliz@fe.up.pt; acbrito@fe.up.pt

KEYWORDS: Flexibility, Agility, Theory, Simulation.

ABSTRACT

This paper presents some lines of thinking related with the establishment of a concrete mathematical basis to measure the ability of enterprises in a supply chain to maintain equilibrium under unexpected variations. It will be argued that *flexibility* will be the most appropriate term to classify the concept behind such idea and that *agility* can be understood as a special case of *flexibility*. Then, a simple theory of *flexibility* directed to demand variations in the supply chain is deduced and commented, as well as some results achieved by dynamic supply chain simulation presented and discussed.

1. Introduction

Flexibility and *agility* are two expressions already well established on the field of Business, even if neither of them had been quantified in the form of a metric that could be applied without misleading by any manager in any part of the world. There is still a diffused and cloudy idea about what in fact each of those terms mean regarding business and management, due to the fact they are multi-variable concepts and normally used to express the same ideas in day-by-day language. Apparently, the handling of such designations depends on the school of thinking as well as on the area of work it is related to. For instance, *flexibility* is more used in manufacturing, where it means something like the ability to answer fast and reliable to needs of changing in the product mixing, while *agility* is most of the times associated with supply chain and thought to be something like the ability of fast response to changes in the overall enterprise (Goldman et al., 1995). Probably, one also can recognise in the usage of such terms a certain pattern of the kind which let Wendy Currie* (2000) say "*the historical analysis of these change management panaceas shows that eventually their popularity and applicability declines and they are replaced by new panaceas which, although labelled differently, are in many ways similar to the predecessors*" [1], while referring to concepts like TQM, JIT, BPR and PI. The fact is that it continues to be extremely difficult to find a single and wide accepted definition for *flexibility* and for *agility*, and even more difficult to find their metrics.

For that reason we decided to start not using such words. Instead we will start using the concept of *elasticity*, which is already deeply treated on Physics by the *Theory of Elasticity*, thus meaning it will be understood by all the scientific community.

Then, the supply chain will be considered a mathematical system with the form of a network with nodes and arcs where materials and information flow. In such a system, each node will exhibit a certain ability to return to its equilibrium concerning the flow traversing it (we will only consider the flow of materials), with such flow being highly dependent on the material sink and source policies governing each node. Obviously, the complexity of the problem will exponentially increase with the increase of the number of nodes in the chain and their own complexity. Based on those aspects will try to establish a practical measure of *flexibility*.

In this paper we first make some comments regarding nomenclature and then we will derive the theoretical basis behind our perspective of measuring *flexibility*. This has naturally resulted from the need of trying to quantify *flexibility* while developing a Dynamic Supply Chain Simulator (Feliz, Brito, 2003)[2] capable of using such metrics. Finally, we present some practical results obtained by simulating a didactic supply chain case.

2. Some Theoretical considerations

Although the *Theory of Elasticity* is somehow a field of knowledge demanding the manipulation of advanced mathematics, mainly differential equations, the approach we present here will be less complex than that and somehow a mixture between signal processing and elasticity. Our expectation is that this will turn these ideas wider applicable in practice and will lead to a simple method of measuring the ability of a system to change state while maintaining the equilibrium.

The theoretical basis of elasticity comes from the problem known in literature as the *Simple Harmonic Oscillator*, which since long time ago is completely described in Physics literature. An excellent and clear description can be found in the book from Murray R. Spiegel, "*Schaum's Outline of Theory and Problems of Theoretical Mechanics with an Introduction to Lagrange's Equations and Hamiltonian Theory*", McGraw-Hill, 1967.

From reading its 4th chapter one can realize that the *elastic forces* are in fact forces directed to the state of equilibrium, but also recognize that those forces are more related with the ability to oscillate around such a state than with the ability of returning to it. *Elasticity*, as a quality derived from the *elastic constants* of the materials, must then be understood as an ability to oscillate and not precisely as an ability to return to the equilibrium. Figure 1 represents a simple system oscillating around an equilibrium state (*Qo*) with a certain period (*To*) of oscillation, period that is dependent on the *elastic constant*. The higher the value of this constant (or the *elasticity*) the higher the rate of oscillation and

* Professor and Director of the Center for Strategic Information Systems at Brunel University, UK

thus the lower the period *To*.

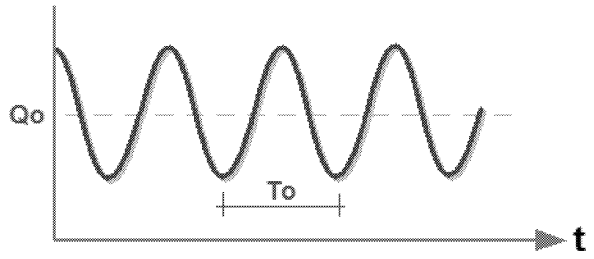


Fig. 1 A pure elastic behaviour

But this of course does not serve our intents, and then we simply refuse to use the term *elasticity* to express the idea of maintaining the equilibrium when a change of state in the supply chain occurs.

However, if we now introduce a certain *force of dumping* in the *Simple Harmonic Oscillator*, such a system will in fact start to exhibit a tendency to return to its state of equilibrium, thus behaving much more similar to the effects we are interested to measure concerning the supply chain responsivity. In figure 2 it is shown the three different modes how this new system can approach the new state. They are known in literature as the *over-dumped* mode, the *critical-dumped* mode, and *under-dumped* mode.

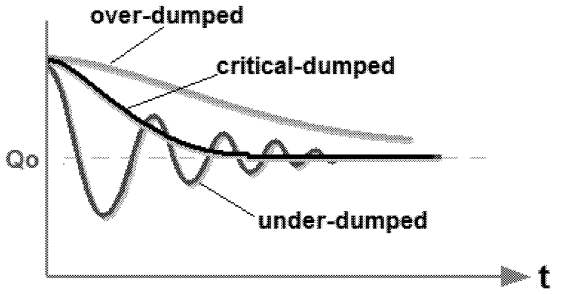


Fig. 2 A dumped elastic behaviour

Once in the *over-dumped* mode the system naturally returns to *Qo* after the initial perturbation is removed, but this time is longer than the optimal response time and thus it will turn the system low. On the other extreme, when *under-dumped*, it will approach *Qo* swinging and thus exhibiting great instability. Anyhow, once *critical-dumped*, the system not only will reach the equilibrium faster but also spending the minimal energy. Systems in such conditions are said to be *well-dumped* or *tuned* to their characteristic frequency of oscillation *fo = 1/To*.

In our point of view, this dumped behaviour already satisfies a certain idea of *flexibility*, for it shows how the system will return to the equilibrium after an instantaneous perturbation. Therefore we are tempted to conclude that a system will be more *flexible* when its *Time to Return to Equilibrium (Tq)* is reduced. At the same time, we conclude that a measure of such a time interval would in fact give us an idea of the *flexibility*.

Another result of this approach is that *agility* could be interpreted as a specific behaviour of *flexibility*, as it points towards the elegancy of movements and thus to the ability of achieving the goals with less energy spending. Thus, *agility* would be understood as the *critical-dumped* condition of *flexibility*. These would lead us to the following statements:

The system is agile when it faster recovers the equilibrium without going into oscillations, implying the minimal energy consumption.

or, saying in another way:

The system is agile when operating in a regime or well-dumped flexibility.

3. The metrics for *flexibility*

In the previous section we have suggested that *flexibility* would be inversely proportional to the *Time to Return to Equilibrium (Tq)*, thus, a simple way to express the *flexibility* is inverting the things and consider:

$$FLEXIBILITY \sim 1 / Tq \quad (1)$$

This expression is equivalent to the *Rate of Returning to Equilibrium*, or, in other words, to the *Ability to Absorb Chocks*.

This is of course a general definition for *flexibility*, and so it must hold for any systems as a basic concept, not only for supply chains. Anyhow, the challenge now is to find a way by which *flexibility* can be measured in practice. For that purpose we will use a simple representation of a state of equilibrium along the time. In fact, if we believe *flexibility* to represent the ability of the system to maintain its equilibrium, then a way to turn "visible" such ability can be plotting over time or the state itself or the function that represents the equilibrium of that state.

With such intents, let us focus the attention on just a node of the supply chain and on the state of its inventory. Obviously, the node will be in equilibrium (in terms of materials flow) when the rates of input and output of materials will be the same. That is, when:

$$R_{out} = R_{in} \quad (2)$$

Being R a quantity of materials per unit of time, a way of calculating these quantities in practice will be to consider the amounts of materials (Q_{in}) entering the node during the time interval (dT_{in}) of two consecutive arrivals from the supplier, and the amount o materials (Q_{out}) leaving the node during the time interval (dT_{out}) between consecutive services to the clients. Equation (2) can then be written as:

$$(Q_{out} / dT_{out}) = (Q_{in} / dT_{in}) \quad (3)$$

or, arranged in another way:

$$(Q_{out} / Q_{in}) \times (dT_{in} / dT_{out}) = 1 = EQUILIBRIUM \quad (4)$$

Anyhow, if we look at the node only at the moments the material arrives from the supplier ($dT_{out} = dT_{in}$), we can say the node will be at perfect equilibrium when the quantity of materials going out equals the quantity of material coming in. In any other circumstances, the node will be out of equilibrium.

Thus, a simple and practical form to represent this equilibrium function is to compute it every time the material arrives, and this will be achieved by counting Q_{in} and Q_{out} during each time interval dT_{in} followed by an operation of division.

It is also important to notice that with this method the function EQUILIBRIUM(t) will be a discrete function "sampled" by the frequency of arrival, as shown in figure 3 (such frequency is given by $f_{in} = 1 / dT_{in}$, as it also can be deduced from the same figure). This important aspect shows that the node is not "during all the time connected" to its suppliers, and for that reason it will face "certain" difficulties on following "certain" demand changes, resulting obvious that systems with longer dT_{in} will have more

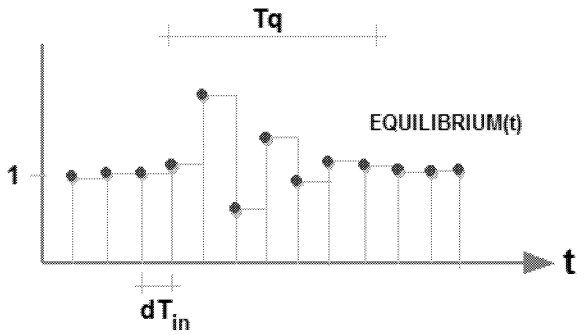


Fig. 3 A perturbation in the EQUILIBRIUM(t) function

difficulties to recover from demand "chocks". We let to the Supply Chain Managers the investigation on how to reduce dT_{in}.

At the moment, and if for simplicity we consider the frequency of arrival to be fixed, the important is to realise that any measure of the *Time to Return to Equilibrium* (*Tq*) by means of this equilibrium function will lead to values multiples of dT_{in}, that is:

$$Tq = n\, dT_{in} \tag{5}$$

Where *n* is an integer assuming values 0, 1, 2, 3, etc., representing the number of arrivals from the supplier. Finally, comparing equation (5) with equation (1) we will deduce:

$$FLEXIBILITY \sim (1/n)\ (1/dT_{in}) = (1/n)\ f_{in} \tag{6}$$

That is, the *flexibility* will be directly proportional to the frequency the material is arriving from the supplier (f_{in}), as well as inversely proportional to the number of such arrivals (*n*).

And this is an interesting result, first because it comes close to a range for this metric, saying the *flexibility* belongs to the interval $[0, f_{in}]$ with an exception of ∞ when the perturbation does not affect at all the equilibrium ($n = 0$), and also because it is in accordance with the "empiric" principle of reducing the reorder quantity in warehouses and the lot quantity in manufacturing systems. Actually, these results fit perfect with the common idea that *a system will get more flexible as its frequency of replacing material rises*. It also results from these considerations that to achieve the optimal *flexibility* (or the state of *agility*) in practice one must adjust (by trials?) f_{in} and the reorder quantities in order to minimize *Tq* without letting the system entering in oscillation. In principle, we believe this can be achieved using the scheme suggested in figure 4 with steps of demand to induce perturbations

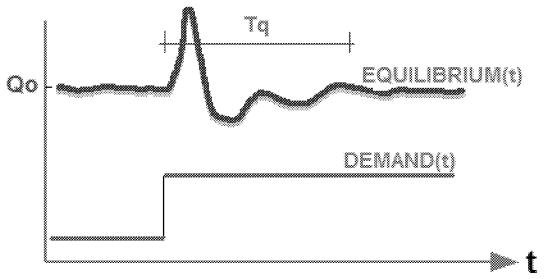


Fig. 4 Using demand steps to measure *flexibility*

in the system, by means of simulation.

Although this seems a reasonable proposal to measure the *flexibility*, it is important to recognize it comes out from a one dimensional analysis of the problem. In fact, it just considers the variable *time* (*t*) and completely ignores the variable *material quantity* (*Q(t)*), and this would mean a system would be the same *flexible* whether going to equilibrium by moving lots of material or by moving only little quantities of it. Of course, it seems reasonable to consider the second case would be more *flexible*, despite the same *Tq*. And this reasoning leads us to a final representation of *flexibility* based on the "energy spent on imbalance".

To do so, lets notice that equation (3) can also be written in the following form:

$$(Q_{out} / dT_{out}) - (Q_{in} / dT_{in}) = 0 \tag{7}$$

what, considering again $dT_{out} = dT_{in}$, leads to:

$$(Q_{out} - Q_{in}) / dT_{in} = 0 \tag{8}$$

which is another way of representing the EQUILIBRIUM function for the inventory and can be though as an "Imbalance Ratio", measured on inventory units (SKU) per unit of time. This already corresponds to a certain "energy" spent per unit of time, and thus gives us an idea of the "*Power Spent on Imbalance*" or *rigidity*, which in fact can be seen as the inverse of *flexibility*. By this definition, the system is more *flexible* when its *rigidity* gets low, and a simple way to measure it will be integrating the EQUILIBRIUM function over time and divide its value by the total time of imbalance. Thus, if for a certain period of time $dT_{in}[j]$ it holds that there was $Q_{in}[j]$ units of material going into the node and $Q_{out}[j]$ units of material going out, the *rigidity* can be calculated as:

$$RIGIDITY = \frac{\sum_j Qin[j] - Qout[j]}{\sum_j dTin[j]} \tag{9}$$

And then,

$$FLEXIBILITY = \frac{1}{RIGIDITY} \tag{10}$$

It is based on this measure the matrix of *rigidity* of a simulated case will be presented later in this paper.

4. Multi-function *flexibility* of a single-node system

Even considering a single node system, it is important to remember we are suggesting only a way to measure the inventory *flexibility* in respect to demand variations. Anyhow, in any general single-node system one have to consider a set of output functions *OUTj(t)* being driven by a set of input functions *INi(t)*, all of them running along a certain independent variable *t*, as depicted in figure 5. That is, we must start with the assumption that any of these functions can be graphically represented in time by some kind of graph.

Observing this figure, it can be easily understood that in

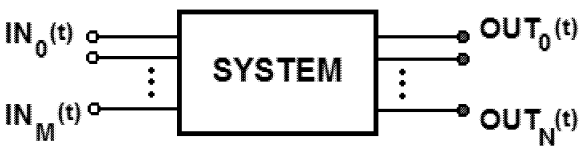


Fig. 5 Generalised multi-function model of a single node system

general one talks about flexibility of the *OUTj(t)* functions in respect to variations of the *INi(t)* functions. If for instance a generic output function *OUTk(t)* is dependent on the generic input functions *INa(t)*, *INb(t)* and *INc(t)*, then one will be automatically able to talk of more than one *flexibility*, as at least the following three obvious types could be considered:

a) *flexibility* of *OUTk(t)* to variations of *INa(t).*

b) *flexibility* of *OUTk(t)* to variations of *INb(t).*

c) *flexibility* of *OUTk(t)* to variations of *INc(t).*

This means in general one will talk about *flexibility* of a *property* in respect to variations of a certain *factor*. If that *property* depends on many *factors*, then many *flexibilities* can be considered for that *property*. To turn the things even more complex, one could also think on the input crossed variations…

So, even being the *flexibility* a simple concept, it usually turns complex if accuracy is important and if the output functions are dependent on more than a single input. This explains why till now many authors try to classify *flexibility* following the different levels strategy proposed by Leslie K. Duclos (2000), where he obviously assumes *flexibility* is a multi-dimensional construct[3].

At the same time, from the point of view presented here, the general *flexibility* of a node will no longer be a *scalar* but a *matrix* connecting the *equilibrium functions of the properties (outputs)* to the *factors (inputs)*. The *flexibility* of a general single node system must then be represented by:

$$FLEXIBILITY = \begin{bmatrix} f_{11} & f_{12} & \cdots & f_{1M} \\ f_{21} & f_{22} & \cdots & f_{2M} \\ \cdots & \cdots & \cdots & \cdots \\ f_{N1} & f_{N2} & \cdots & f_{NM} \end{bmatrix} \qquad (11)$$

Where f_{ij} is the individual *flexibility* of the *OUTi(t) property* in respect to variations of the *INj(t) factor*. Each of the f_{ij} values can be quantified as long as for each *OUTi(t)* one will establish a way to compute the respective EQUILIBRIUM function and a way of exciting it with the appropriate *input step*. In each case the big challenge is to define the EQUILIBRIUM function. What is this function for the *costs*, for example? Is it easy to describe? And for many other terms of the Supply Chain?

5. N nodes *flexibility* (the supply chain flexibility)

But practical systems are most of the times more complex than a single node system, and when talking of Supply Chains the complexity usually rises. We have shown in the previous section that in general the *flexibility* of a node would be represented by a matrix, from what follows that the *flexibility* of a group of nodes would be a matrix of matrices, and thus something too complex to easily be handled in practice. However, if we are only interested on a particular component of its output vector, that is, on just one output function (the *inventory*, for example) and consider it only depends on a variable (the *demand*, for example), the node *flexibility* already can be reduced to an easier measurable *scalar* quantity. For simplicity, we will assume each node is represented by such a simplistic model.

In that case, the Supply Chain is then reduced to a collection of *inventory* nodes, each one with a certain *flexibility* dependent on its previous element (the supplier) as well as on the reorder quantities. Therefore, the *flexibility* of the overall Supply Chain (in respect to a certain *demand* pattern) is again a matrix where the individual flexibilities will be plotted.

As an example to compute some practical results we will use the Supply Chain structure drawn in figure 6, which represents the "*Cranfield Blocks Game*"[4], a management game developed by Richard Saw (2002)[5].

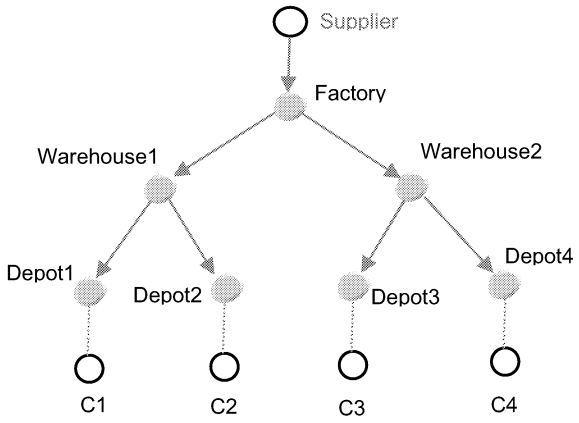


Fig. 6 Structure of the "*Cranfield Blocks Game*"

This is a seven node Supply Chain (from *Depot1* to the *Factory*) where the *direct demand* (*inputs*) from *customers* (*Cj*) is injected at the depots level, and the *Supplier* is considered an infinite source of materials. For simplicity, only one product is being considered.

From what concerns our interests this can be thought of a system with four (4) inputs driving certain output functions at each of its seven (7) nodes. Thus, by means of injecting steps of demand at each input, one will be in principle able to measure how each node functions (like inventory, costs, etc.) will respond to such "shocks", and through such measure represent the correspondent overall *flexibility* matrix.

In general, a simple way to write down this matrix will be plotting the individual flexibilities in columns ordered by increasing level of the facilities on the chain, as next figure suggests, which fast let us to have an idea on how the chain is susceptible to input "chocks", as it will be shown in a simple example presented soon.

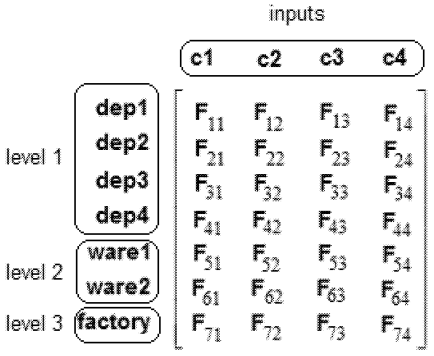


Fig. 7 General *flexibility* matrix for the "*Cranfield Blocks Game*"

Following this representation, the *flexibility* of the chain will be the *flexibility* of its "arms" connecting the *last customers* to the *primer suppliers*. Thus, it is somehow understandable that a particular Supply Chain would show high flexibility on serving certain customers and at the same time exhibit disastrous

flexibility on serving others. A nice result from this approach, however, is that "*the flexibility of an arm will be extremely dependent on the flexibility of its slower node*", since the speed on a chain is the speed of its slowest element, what means it can sometimes be very important to ensure a good cooperation between partners in the Supply Chain.

5.1 *The Inventory-flexibility-to-demand*

Following these ideas, we now present the determination of the *Inventory-rigidity-to-demand* matrix for the "*Cranfield Blocks Game*", achieved by simulation. For convenience, we will use the *rigidity* representation, as it always deals with finite quantities.

In this case, all the facilities on the game have been configured to use the *(r, Q)* stock reorder policy and the demand at each *Depot* have been initially kept constant at the average value the manual version of the game states. Anyhow, at a certain moment of the simulation process, the demand of a certain *Depot* received a sudden increment on its value, and the new situation was maintained till the end of the simulation. This procedure was done in each of the *Depots* separately. In each case, it was measured at each node of the Supply Chain the *rigidity*, applying the equation (9), and then the matrix represented. Also to simplify the read of the results the null values have been substituted by dashed lines in the matrix. For the simple propose of this paper, each simulation have been executed with only 3 replications, but the averages and irrespective uncertainties have been computed for 95% confidence as if more than 20 samples would have been used.

At the figure 8 is represented how the stock level of *Depot1* behaved when subjected to a demand step of 2.5 times its usual average value. Notice that even if the system is answering good to such requirements there was introduced on it a certain instability that did not exist before.

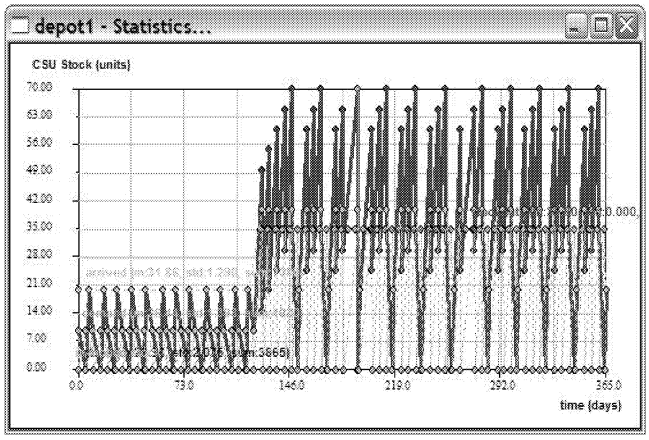


Fig. 8 Behaviour of *Depot1* stock level in reaction to a demand step

In the next figure (fig. 9) is represented the behavior of the EQUILIBRIUM(t) function for the same *Depot1* as it is described by equation (4). We opted to use this form of presenting the equilibrium to the user once it is more interesting to plot graphically than the one resulting from equation (7). From this figure is already evident that a certain instability have been introduced due to the abruptly change of demand. This instability, however, is mainly due to the lack of an integer relation between the new demand and the quantity ordered to the supplier. By readapting this quantity one could easily reduce this kind of instability. This also lets us induce that different reorder policies can conduct to different *flexibility* values.

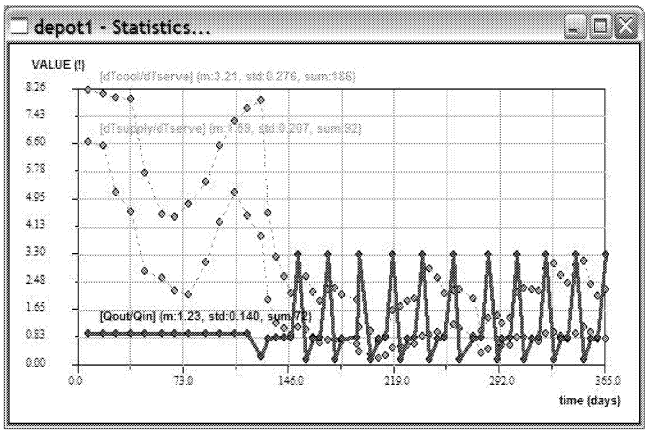


Fig. 9 EQUILIBRIUM(t) function of *Depot1* under a demand step (bold line)

It comes also obvious from this figure that the "*Time to Recover Equilibrium*" (*Tq*) would be difficult to compute in cases like this, once the system does not stop to oscillate. Nevertheless, to measure the RIGIDITY using the second approach (as it is used here) eliminates this problem and stabilizes the measure independently the time of the window of observation.

And finally, in the figure 10 we present the RIGIDITY matrix for this case, resulting from the four simulation processes, each one of three runs. Al the values presented are in SKUs/day.

		inputs			
		c1	c2	c3	c4
level 1	dep1	4.6 ± 0.3	----	----	----
	dep2	----	8.5 ± 0.8	----	----
	dep3	----	----	13.6 ± 0.6	----
	dep4	----	----	----	6.7 ± 9.0
level 2	ware1	1.6 ± 0.0	3.9 ± 0.6	----	----
	ware2	5.0 ± 0.4	5.0 ± 0.3	7.4 ± 0.7	6.5 ± 6.4
level 3	factory	4.6 ± 2.4	5.2 ± 1.3	4.6 ± 0.7	5.6 ± 1.6

Fig. 10 General RIGIDITY matrix for the "*Cranfield Blocks Game*"

It is of course important to notice that this matrix is valid for certain specific conditions, and so those conditions must be well specified before interpreting the data. In this case, the conditions could be roughly represented by:

All Step Demand Amplitudes = 2.5 x (usual average demand).
All Reorder Policies = (r=ro, Q=Qo), as the game stated.
Each facility fleet = 1 vehicle Van.
All Depots start from EQUILIBRIUM.
Etc.

5.2 *Interpreting the matrix*

It is now clear from this matrix that a step on demand of amplitude 2.5x on the *Depot1* will let us observe a "*Power Spent on Imbalance*" of 4.6 SKU/day in this facility, as well as 1.6 SKU/day at the *Warehouse1*, 5.0 SKU/day at the *Warehouse2* and 4.6 SKU/day at the *Factory*. For instance, it can also be expected that the same kind of step at the *Depot3* will lead to a much higher *rigidity* at that facility as well as to a higher influence on the *Warehouse2*, which will exhibit 7.4 SKU/day of imbalance. Anyhow, such a step on *Depot3* will never influence *Warehouse1* behavior. And so on…

Another interesting aspect one can infer from this matrix is that probably the imbalance of 5.0 SKU/day shown by *Warehouse2* is not motivated by the steps, as it maintains almost constant on the cases C1 and C2. This is probably a residual imbalance due to the natural uncertainty on the *Warehouse2* demand due to unsynchronized *Depot3* and *Depot4* reordering moments.

A last interesting aspect coming out from this data is that it was impossible to obtain a reliable value on the C4 case, as the uncertainties are of the same order of the averages. And in fact, as the next figure shows, it was observed such a high *stockout ratio* on *Depot4* that hardly it could lead to reliable values for the *rigidity*.

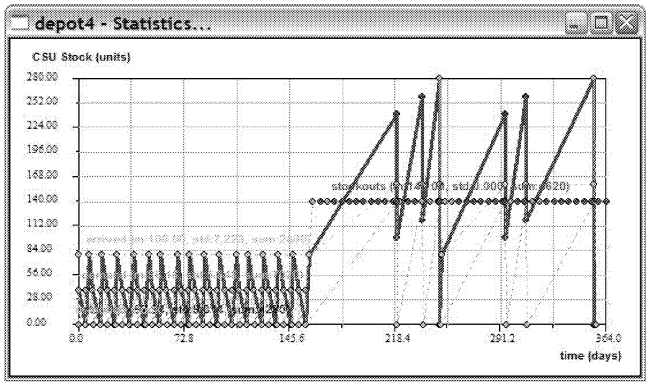


Fig. 11 Stock level and stockouts at the *Depot4* facility

This also have been confirmed by the "Satisfaction ratio" observed at the customer C4, presented in the figure 12, which have dramatically rolled down after the "chock" of demand.

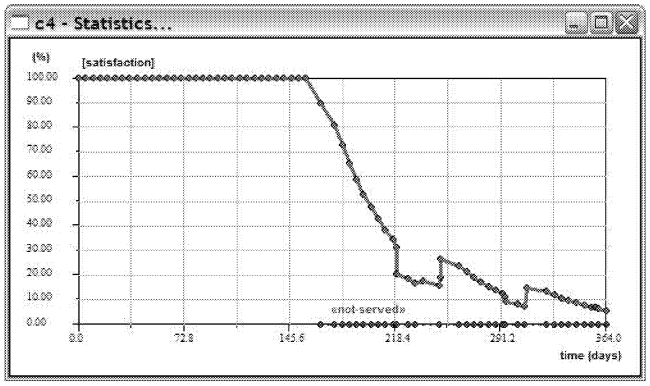


Fig. 12 "Satisfaction Ratio" at the C4 customer

Based on this case, a deeper study concerning the *flexibility* dependency on reorder policies, reorder levels and step amplitudes will be published in a paper to come.

6. CONCLUSIONS

This results, achieved with the help of simulation, let us conclude that the proposed methodology to measure *rigidity* in a Supply Chain is feasible and the results can be somehow easily interpreted, therefore conducting to the possibility of applying a reasoning and, trough it, to support decisions.

At the same time, we also conclude that *flexibility* depends more on the suppliers than on the customers, which automatically remind us the idea of good *cooperation* with suppliers. It seems also natural *flexibility* results form an interrelation from where concepts as *cooperation*, *visibility* and *trust* automatically flow. At the same time, it have also been shown that *flexibility* can depend on the resources used and on how they are used, like the number of vehicles dedicated to delivery, for example, or the model used for reordering the material.

Of course that it is obvious the *flexibility* can be potentially made higher if working with more than one supplier, using redundancy, the usual way to think on flexibility, but that does not give the manager any quantitative values for reasoning, as it is no more than a holistic proposal, forgetting that more than a supplier can imply serious difficulties in relationship, due to concurrence.

Finally, one must remember that the present results have been obtained considering only "positive" steps of demand, that is, incrementing it, and nothing have been said for the opposite case, when there is a sudden decrement on demand. Anyhow, as the usual Supply Chain is mainly a flow of products in only one way, it results a natural true the best approach on theses cases it is to maintain the minimum stock possible, which in the limit would be a single unit of material (JIT).

Future research on this matter could include studies about other kind of Supply Chain structures, not only didactic but also representing real systems, and, using the kind of analysis we presented here, enrich the knowledge about the idea of *flexibility* with the final intent to turn it more scientific that what in fact seems at the moment. Testes could be made about *flexibility* on many aspects of the Supply Chain, as for instance with different reorder policies, or demand amplitudes, as well as with diverse resources and policies related with transportation, or even other. This seems, in fact, an open field for tests using simulation.

References:

[1] Wendy Currie, Vlatka Hlupie, "*Simulation Modeling: the link between change management panaceas*", 2000 Winter Simulation Conference, J.A.Joines et al. eds. 2000.

[2] J. Manuel Feliz-Teixeira, António E. S. Carvalho Brito, "*An Approach for Dynamic Supply Chain Modelling*", The 2003 European Simulation and Modelling Conference, Naples, Italy, October 27-29, 2003, pp.132-139.

[3] Leslie K. Duclos, et all. "*A Conceptual Model of Supply Chain Flexibility*", University of Northern Iowa, 2000.

[4] This game is also described at J. Manuel Feliz-Teixeira, António E. S. Carvalho Brito, "*Distributed Application for Supply Chain Management Training*", GEIN - Faculdade de Engenharia da Universidade do Porto, Portugal, 2003.

[5] Richard Saw, "*Cranfield Blocks Game*", private communication, Centre for Logistics and Supply Chain Management (CSCM), University of Cranfield, 2002.

- Tsourveloudis, N., "*On the Measurement of Agility in Manufactoring Systems* ", University of Southwestern Louisiana, The Center for Advanced Computer Studies, Robotics and Automation Laboratory, USA, 2001(?).

- Harding, L., "*QRM: An Enabler on the Road to Agility*", Control, 2002, pp. 15-18.

- Bal, J., Wilding, R., & Gundry, J. "*Virtual Teaming in the Agile Supply Chain*", Journal of Logistics Management, 1999, 10(2), pp. 71-82.

- Christopher, M., "*Creating the Agile Supply Chain* ", Cranfield School of Management, U. K, 2000.

- Hoek, R., "*Towards the Agile Supply Chain* ", (PowerPoint Presentation). Washington DC: Cranfield School of Management, 2001.

- Lambert, D. M., & Pohlen, T. L., "*Supply Chain Metrics*", The International Journal of Logistics Management, 2001, 12(1), pp. 1-19.

SIMULATION TOOLS

EXTENDING SCICOS FROM SYSTEM TO COMPONENT LEVEL SIMULATION

Masoud Najafi, Azzedine Azil, Ramine Nikoukhah

INRIA*
Rocquencourt, BP 105
78153, Le Chesnay cedex
France

KEYWORDS Dynamic system simulation, causal and acausal systems, Hybrid systems, DAE, Simulation software, Numerical solver.

ABSTARCT

The capabilities of the hybrid dynamic system simulator Scicos has been extended to allow natural modeling of physical components which a-priori have no causality structures. This paper discusses these new features and the way they have been implemented in Scicos.

INTRODUCTION

Scicos is a Scilab toolbox[1] [3, 4] for modeling and simulation of dynamical systems. It provides a block diagram graphical editor for the construction of complex dynamical systems including hybrid systems [14, 15, 16] . A new extension of Scicos allows the use of "implicit blocks" in the construction of models. The ports of an implicit block are not labeled, a priori, as inputs and outputs, and the block imposes a dynamical constraint on the values on its ports, contrary to regular blocks which compute explicitly their outputs as functions of their inputs. Implicit blocks are essential for constructing models which include physical components such as resistors, capacitors, etc., in electricity, or pipes, nozzles, etc., in hydraulics. They also arise in many other areas such as mechanics and thermodynamics.

*The french national institute for research in computer science and control

[1]Scilab is a free open-source software for scientific computation, see www.scilab.org and www.scicos.org.

CAUSAL VS ACAUSAL MODELING

Modular model construction can be classified in two major categories: causal and acausal. To make an analogy with computer programming languages, causal modeling corresponds to the use of assignment statements where the right-hand sides of the equations are evaluated and the result of the evaluation is assigned to the variables on the left-hand side of the equations. This corresponds closely to the evaluation of the outputs based on the inputs in a causal module. Acausal modeling on the other hand is closer to *equations* in the mathematical sense of the term. For an equation in a set of mathematical equations, it is not possible to classify (at least a-priori) its variables as inputs and outputs. In this case, for constructing the solution to the set of equations, the choice of the variable which should be computed in terms of the rest of the variables, in one particular equation, depends on the complete set of equations. In other words, a mathematical equation contains potentially a number of assignment statements; the choice of the statement which should be used for constructing the solution of the complete problem depends on all the equations.

An acausal module is like a mathematical equation. It has ports of communication with other modules but these ports are not labeled a-priori as inputs and outputs.

Most physical components are more naturally modeled as acausal modules simply because physical laws are expressed in terms of mathematical equations. Consider a resistor in an electrical circuit. If the resistor is to be used as a module, it can only be an acausal module because depending on the way the resistor is used, the input can be a current and the output a voltage, or the input a voltage and the output a current.

In most modeling and simulation software, only causal modules (blocks) are considered. This situation is not only easier to implement but it provides

a stronger notion of modularity which corresponds closely to the notion *subsystem* used by systems and control engineers. This is the reason why such way of modeling is referred to as *system level modeling*, as opposed to *component level modeling* where acausal modules (blocks) are used.

It is in general possible to convert a component level model into a system level model by rewriting the equations and finding the appropriate causality structure in each module, but this task is time consuming, and error prone. Furthermore the resulting (system level) model has very little resemblance with the original model making subsequent modifications difficult to realize. Indeed, a small change in the original model may require a complete redesign of the system level model.

To illustrate the difference between component level and system level modeling, consider the simple electrical circuit shown in Figure 1.

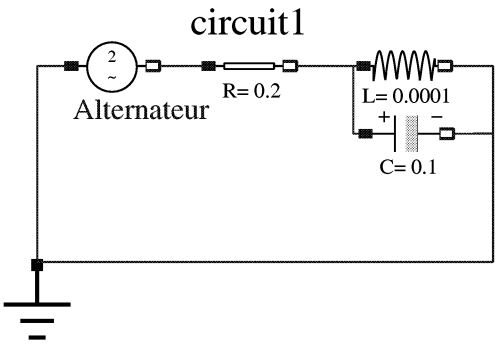


Figure 1: The original electrical system to be modeled

The circuit in Figure 1 clearly corresponds to a component level model. This is what an electrical engineer would like to work with. Using elementary models for each electrical component, we obtain a corresponding equivalent system level model depicted in Figure 2. We see clearly an important drawback in using such a model: there is no one to one correspondence between original circuit elements and the blocks in the diagram. It is not difficult to see that for example replacing the resistor with a diode, would require starting over the construction of the system level model. This is an important point because even for this very simple example, the analysis to convert the intuitive physical model to a system level model is non-trivial and cumbersome.

It turns out however that in many situations, in particular in systems and control applications, the use of causal modules is very useful. As an example in figure 3, the F14 flight control system has been illustrated. This system is composed of several parts such as Actuator, aircraft, control, feedback and some other dy-

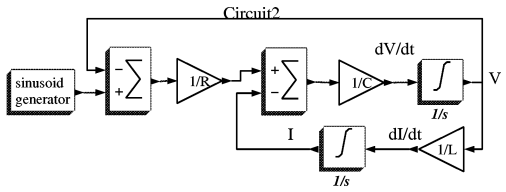


Figure 2: The system-level (causal) implementation of model depicted in figure 1

namics. In such control systems it is much easier for the control engineer to have a system level model to use for the design of the controller and to optimize the control parameters.

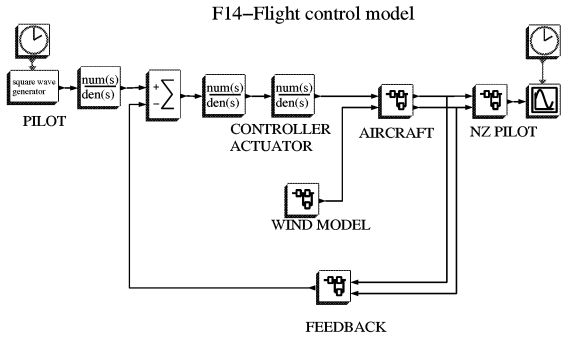


Figure 3: A system level modeling and design

Even though causal models can in some sense be considered as special cases of acausal models, the restrictions imposed by causality implies that the behavior of the module can be implemented using an almost block box program which evaluates its outputs as a function of its inputs. The implementation of the behavior of the module in the acausal case is more complicated and requires either the construction of multiple causal modules or a formal description. In the latter case, the construction of the complete model requires formal manipulation of equations obtained from the description of acausal modules and their connection topology. It is for this reason that in Scicos we allow the co-existence of both type of modules: causal and acausal.

MODELICA: A STANDARD IN COMPONENT LEVEL MODELING

To extend the capacity of Scicos to allow component level modeling, we needed a language for describing the algebraic-differential constraints imposed on the

input/outputs of the acausal blocks. We found Modelica [5] to be an excellent choice.

Modelica is primarily a modeling language, sometimes called hardware description language, for specifying mathematical models of physical systems, in particular for the purpose of computer simulation. Modelica is a modern object-oriented programming language based on equations instead of assignment statements (it can thus be classified as a declarative language)[8]. This makes it particularly useful for the purpose of acausal modeling, and this is not a surprise because Modelica is developed exactly for such applications. As an example the electrical circuit in figure 1 can be expressed in modelica as follows:

```
class imppart_circuit1
  parameter Real P1;
  parameter Real P2;
  parameter Real P3;
  parameter Real P4;
  parameter Real P5;
  parameter Real P6;
  Resistor   B1(R=P1);
  Inductor   B2(L=P2);
  Capacitor  B3(C=P3, v(start=P4));
  Ground     B4;
  VsourceAC  B5(VA=P5, f=P6);
equation
  connect (B2.p,B1.n);
  connect (B3.p,B1.n);
  connect (B5.p,B4.p);
  connect (B2.n,B4.p);
  connect (B3.n,B4.p);
  connect (B5.n,B1.p);
end imppart_circuit1;
```

where each component has been defined in the Modelica's library. For example here is the model of a simple Inductor.

```
class Inductor "Ideal Inductor"
      Pin p, n;
      Real v;
      Real i;
      parameter Real L "Inductance";
equation
      v = L*der(i);
      v = p.v - n.v;
      0 = p.i + n.i;
      i = p.i;
end Inductor;
```

We are not going to present Modelica here in this paper, interested reader can find all the necessary information from `www.modelica.org` [5].

Modelica language is very rich and has the capacity also to handle discrete time systems. Recently, the language has been enriched to allow modeling of finite state machines as well. Our use of Modelica however, at least for the moment, is limited to the continuous-time part. We think that, except for very special cases, modeling discrete events can be done more naturally at the system level, where Scicos provided already the necessary functionalities for modeling and simulation. We thus have found that the best compromise is to allow the usage of acausal blocks in the existing Scicos environment while keeping the traditional causal blocks the way they were. Both types of blocks of course should be usable in the construction of diagrams.

Since the information available on the links connecting acausal models corresponds in general to physical quantities such as (voltage, current), (flow,pressure), etc, and are very different from information on links connecting causal models which transport signals, even a simple derivation placed on a link has a very different significance in two cases: a link that splits in two in the causal setting simply means that the information is duplicated whereas in the acausal setting, say in an electrical circuit, such a split must be implement using the Kirchof's voltage and current laws. Clearly it would be meaningless to allow the direct connection of a port of an acausal model to that of a causal model. That is why, special blocks were introduced in Scicos to be used as interface between the causal and the acausal world. Such blocks which have very clear physical significance, may have input ports, output ports *and* "acausal ports". An example of such a block is a voltmeter. This block has two acausal ports; physically they correspond to the place where the two wires are to be connected. But it also has an output where the measured value of the voltage is sent out of the block. In general, sensors and actuators make the interface between the component level part of the model and system level part.

Consider the diagram in Figure 4. Here we have a fluid level control problem. Component level modeling is used to model the fluid dynamics in a very natural way using two sources, a regulated valve, a container and a tube. The container has a built-in level sensor which makes the interface with the system level part of the system, similarly the valve is regulated through an input signal from the system level part. The controller and the display mechanism are implemented at the system level.

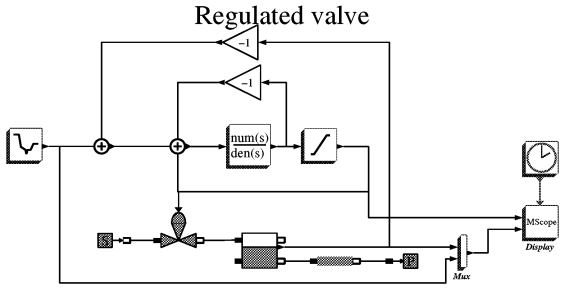


Figure 4: Scicos diagram containing both types of blocks.

IMPLEMENTATION IN SCICOS

The way acausal blocks are handled in Scicos is as follows. First, all the acausal blocks (including the interfacing blocks) are grouped and replaced with a single causal block. This operation is of course totally transparent to the user but it is illustrated in Figure 5 for the diagram in Figure 4. Once all the acausal blocks are aggregated, only the explicit ports of the interfacing functions which are at the boundaries of the two worlds are left visible and thus the result is a causal block.

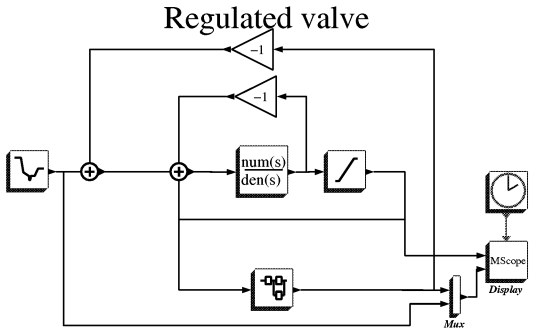


Figure 5: Acausal part reduced to a single block.

The behavior of this new causal block is defined by the Modelica program resulting from the Modelica programs of each of the acausal blocks and the way they are interconnected. This program is automatically generated, compiled and incrementally linked to Scilab making it usable in Scicos.

This new block is almost a standard Scicos block and can be treated like any other Scicos block so that the way compilation and simulation were performed can be carried out as before. There is however one small difficulty. So far in Scicos, not only we had assumed that the blocks where causal but also that if a block had an internal state, the dynamics of this system were explicit. If we note the inputs, outputs and the state of the block respectively by u, y and x, this condition amounts to imposing the following structure on the internal dynamics of the block

$$\begin{aligned} \dot{x} &= f(t,x,u) \\ y &= g(t,x,u) \end{aligned}$$

But the equations governing the new block can very well not be explicit. In many applications in particular in mechanics and electronics, we end up with differential-algebraic systems of equations which cannot always be converted into explicit form.

$$\begin{aligned} 0 &= f(t,\dot{x},x,u) \\ y &= g(t,x,u) \end{aligned}$$

Allowing internal implicit dynamics did not change the overall design of the Scicos compiler but did require the implementation of a DAE solver; DASKR was chosen [10, 11, 13].

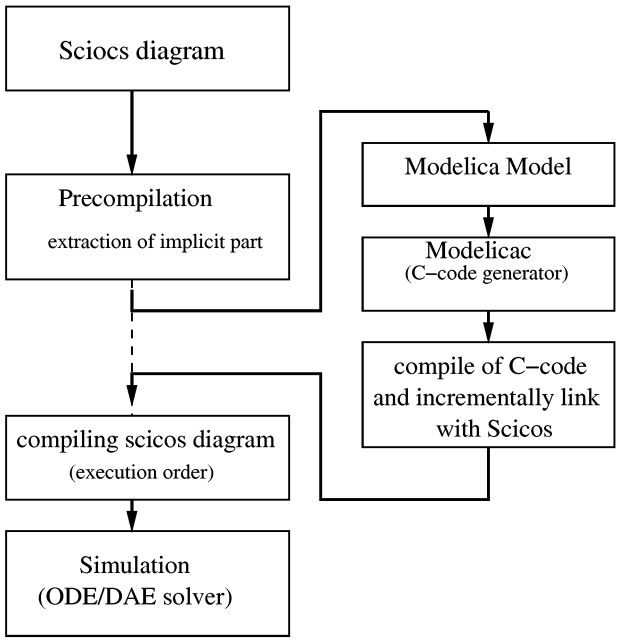


Figure 6: system flowchart

Once a Scicos diagram edited, to simulate this system several operation are performed in an automatic and transparent way. First, the precompiler removes the super blocks to obtain a flat diagram. Then if model contains implicit blocks representing components, precompiler extracts them and their connecting links to

generate a modelica model. The modelica model is then saved in a temporary file to be treated by Modelicac which is Scicos's code generator. Modelicac translates the modelica file into appropriate Scicos C-code. Finally this code is compiled and incrementally linked with Scilab as a new block. The new block is connected to the rest of original Scicos diagram in place of removed parts. After these operations, Scicos diagram is a standard one and is compiled and simulated as usual.

CONCLUSION

In this paper, system level and component level simulations as two approaches for modeling have been presented. Then the needs to simulate the system at component level have been shown. And finally the implementation of component based simulation in Scicos has been discussed.

References

[1] K. E. Brenan, S. L. Campbell, and L. R. Petzold, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*, SIAM publication, Philadelphia, 1996.

[2] P. N. Brown, A. C. Hindmarsh, and L. R. Petzold, "Consistent initial condition calculation for differential-algebraic systems", *SIAM J. Sci. Comp.*, no. 19, 1998.

[3] C. Bunks, J. P. Chancelier, F. Delebecque, C. Gomez(ed.), M. Goursat, R. Nikoukhah and S. Steer, *Engineering and Scientific Computing with Scilab*, Birkhauser, 1999.

[4] J. P. Chancelier, F. Delebecque, C. Gomez, M. Goursat, R. Nikoukhah, and S. Steer, *An introduction to Scilab*, Springer-Verlag, 2002.

[5] P.Frintzson, *Object oriented Modeling and simulation with modelica*, Wiley interscinece, 2004.

[6] A. C. Hindmarsh, "LSODE and LSODI, Two New Initial Value Ordinary Differential Equation Solvers", *ACM-Signum Newsletter*, Vol. 15, 1980, pp. 10–11.

[7] A. Kroner, W. Marquardt, E.D. Gilles, "Computing consistent initial conditions for differential-algebraic equations", *ESCAPE-1 Comput. Chem. Engng* , 1992, Elsinore, Denmark, 16 (1992), Suppl., S131–S138.

[8] S.E. Mattsson, H. Elmqvist, M. Otter, and H. Olsson, "Initialization of Hybrid Differential-Algebraic Equations in Modelica 2.0", *2nd Inter. Modelica Conference 2002*, Dynasim AB, Sweden and DLR Oberpfaffenhofen, Germany, 2002, pp. 9–15.

[9] R. Nikoukhah and S. Steer, "Conditioning in hybrid system formalism", *International Conference Automation of Mixed Processes:A D P M Hybrid Dynamic Systems*, Dortmund, Germany, 2000.

[10] L. R. Petzold, "Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations", *SIAM J. Sci. Stat. Comput.*, No. 4, 1983.

[11] L. R. Petzold. "A Description of DASSL: A Differential/Algebraic System Solver", *In Proceedings of the 10th IMACS World Congress*, Montreal, 1982, pp. 8-13.

[12] A. Benveniste, *Compositional and Uniform Modeling of Hybrid Systems*, IEEE Trans. Automat. Control, AC-43, 1998.

[13] P. N. Brown, A. C. Hindmarsh, and L. R. Petzold, *Consistent Initial Condition Calculation for Differential-Algebraic Systems*, SIAM J. SCI. COMP., NO. 19, 1998.

[14] R. NIKOUKHAH AND S. STEER,*Scicos a dynamic system builder and simulator*, IEEE INTERNATIONAL CONFERENCE ON CACSD, DEARBORN, MICHIGAN, 1996.

[15] R. NIKOUKHAH AND S. STEER, *Hybrid systems: modeling and simulation*, COSY: MATHEMATICAL MODELLING OF COMPLEX SYSTEM, LUND, SWEDEN, SEPT. 1996.

[16] R. NIKOUKHAH AND S. STEER, *Scicos: A Dynamic System Builder and Simulator, User's Guide - Version 1.0*, INRIA TECHNICAL REPORT, RT-0207, JUNE 1997.

AN I/O SIMULATOR FOR WINDOWS SYSTEMS

Jalil Boukhobza
Claude Timsit
PRiSM Laboratory
Versailles Saint Quentin University,
45, Avenue des Etats Unis 78000 Versailles
France
E-mail : {jalil.boukhobza, claude.timsit}@prism.uvsq.fr

KEYWORDS

I/O performance, Discrete simulation, File system cache, Disk, Operating systems.

ABSTRACT

We present in this paper an I/O system simulator for the Windows environment. This simulator takes into account all of the physical disk parameters as well as the behavior of the OS (file system cache, access modes, etc). All of this architecture is driven by a highly configurable I/O workload generator. The purpose of the generator is to allow the application developer to model and tune his application in terms of number of I/Os, request sizes, access modes, inter arrival times, etc.
In addition to this simulator, we developed a storage parameter identification and extraction tool. Such a tool is crucial to provide the user with all the low-level and complex characteristics of his storage system. Once this is done, all these characteristics are to be injected in the simulator to allow a fine simulation of the system.
This yields to a very versatile model which is able to simulate any sort of program behavior with a very good approximation, request by request. The validation on three different machines giving less than 3% variation between real and simulated performance is shown in this paper.

INTRODUCTION

Every Windows I/O critical application designer needs to know the behavior of his application on different storage architectures and for a given file access strategy before and during the whole application's life cycle (video on demand for example (Chan and Tobagi 2003)). Experiments show that I/O performance of Windows systems depends highly on the implemented strategy, and while a performance ratio of three is frequently observed, a ratio of up to ten has been noticed in some cases (Boukhobza 2004).
This paper presents a Windows I/O system and device simulator. This simulator is intended to help the application developer to predict and optimize the designed application I/O performance for a given system architecture in terms of disk configuration (track size, seek times, zoning, head switch time, cache prefetching algorithms for write and read operations, etc), interface definition (transfer rates, sharing of the I/O bus, etc), file system cache strategies for prefetching, write back and write through policies, memory copy transfer rates, etc. This tool has been developed relying on the experience earned during exhaustive measurements of Windows IO system as a black-box (Boukhobza 2004), we finely analyzed and determined all the main parameters involved in IO performance. These measurements show the same file system cache behavior for the Windows systems whatever the file system (FAT32, NTFS) and the used operating system (all versions of Windows 2000 and Windows XP). This simulator has been validated on many different configurations; three of them will be shown in the present document.
The most common way for an application (on Windows OS) to access a file is by using the I/O Win32 functions *CreateFile()*, *ReadFile()* and *WriteFile()*. The *CreateFile()* function opens and creates files. We can specify in this function with the help of some flags the way data are accessed. This helps the file system cache to optimize the throughput by choosing, for example, the appropriate system prefetching algorithms. For the read operation, the three most important modes that have been analyzed:

- The normal mode, which is the default mode, is activated by the FILE_ATTRIBUTE_NORMAL flag.
- The sequential mode (for sequential accesses) is acti-

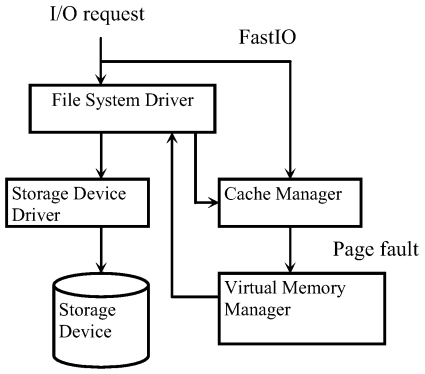


Figure 1: Simplified I/O Request Flow in the Windows System : when accessing a file on Windows systems in the "no buffer" mode, the request is sent to the file system driver then to the storage device driver that forwards it to the disk controller. For the other modes (sequential and normal), the system tests the possibility to send a "fastio" that is a request that copies directly the data if they are present in the file system cache (under some conditions). If a "fastio" is not possible, an IRP (I/O Request Packet) is generated and sent to the cache manager. If the data are not present, a page default is generated by the virtual memory manager and the IRP is forwarded to the device driver to load the data (for reads) (Solomon and Russinovich 2000) (Oney 1999)

vated by the FILE_FLAG_SEQUENTIAL_SCAN flag.
- The "no buffer" mode is activated by the FILE_FLAG_NO_BUFFERING flag.

Both the normal and the sequential modes use the file system cache (see Figure 1). However, the implemented prefetching algorithms are not the same. The "no buffer" mode does not use the file system cache mechanism.
For the write operation, three behaviors were noticed:
- In the "no buffer" mode, the file system cache is not used and the disk cache is employed with a lazy write algorithm (specific to the disk).
- The normal and sequential modes behave the same for the write operations. They both use lazy write algorithms at two levels of the storage architecture: the file system cache (OS dependent) and the disk cache (disk dependent).
- The write through mode is activated by adding the FILE_FLAG_WRITE_THROUGH flag, both the file system cache and the disk cache use no lazy write algorithm.

The exhaustive reverse engineering work that we have done on Windows to extract the file system cache prefetching, lazy write and write through algorithms using micro-benchmarking is the core of our effort. It allowed us to build the heart of the simulator. Two parts can be distinguished in a given Windows storage architecture:
- The OS behavior in terms of cache strategies, I/O scheduling, request sequencing (Thekkath et al. 1994), etc. This part was found to be similar for all the tested Windows versions (Boukhobza 2004).
- The disk drive behavior: in terms of disk prefetching algorithms, lazy and write through algorithms, mechanical activity, etc. This part depends on the plugged disk. To determine the simulation parameters specific to the used storage system, we have developed a parameter extraction tool. This allows the simulator user to get a complete description of the storage system to insert in the simulator.

A very flexible I/O generator has been implemented. It allows the application designer to choose the number of independent workload generators, each of them characterized by request inter arrival times, access modes, position of data space to read or write, etc.

THE WINDOWS I/O SYSTEM SIMULATOR ARCHITECTURE

The Windows I/O system performance simulator has been developed under the OMNET++ discrete event system simulation environment (Varga 2003). It has been chosen because of the facilities it provides, the "easy to use" visual interface (that makes debugging very easy) and the efficient result collecting methods. The developed simulator consists of a set of modules (see Figure 2) communicating via message passing mechanisms:
- The disk module simulates all of the mechanical latencies occurring during I/O access taking into account rotational speed, number of cylinders, number of platters, the zoning information, the different seek times (relative to the number of cylinders to cross), the head switching time, track and cylinder skew, etc., in addition to the mapping of data on the disk.
- The disk buffer module in which is implemented the disk cache behavior with the different algorithms and parameters: capacity, number of segments and their size, updating policies, lazy write and prefetching algorithms.
- The I/O scheduler module is a queue containing the requests that have not yet been served. Many scheduling policies have been implemented: FCFS, SCAN, LOOK, (Worthington et al. 1994) etc.
- The system process and the file system cache modules simulate the file system cache activity of the tested Windows systems (Win2000 and Win XP on both FAT32 and NTFS). This cache launches for the read operation prefetchings for the normal and sequential modes. It follows different algorithms in both modes. For instance, one of the Windows file system cache characteristic for example is that it works with 64KB blocks, meaning that each request of size greater than 64KB is divided in several subrequests. Another important point is that even if there is no inter arrival times between requests, the file system cache does interact with the application requests and in some cases modifies the sequence of the requests as seen by the disk.

This module implements the Windows file system cache read ahead, lazy write, write through algorithms, request subdivision and concatenation, etc.
- The memory and application process modules simulate the application process behavior. When a request is sent by a given application, it can be taken into account by the system process that will satisfy it and then copy data from/into the kernel memory space (file system cache) into/from the user memory space in case of a read/write.
- The I/O generators are the starting point of a simulation. They present a highly flexible model in which we can easily define the number of generators that model the target application to simulate: the inter arrival times, the request sizes, the operation to perform and the logical addresses to access by each generator. The implementation of different distributions for each of the defined parameters is provided thanks to OMNET++ (uniform, exponential, normal, binomial, poisson, etc.).

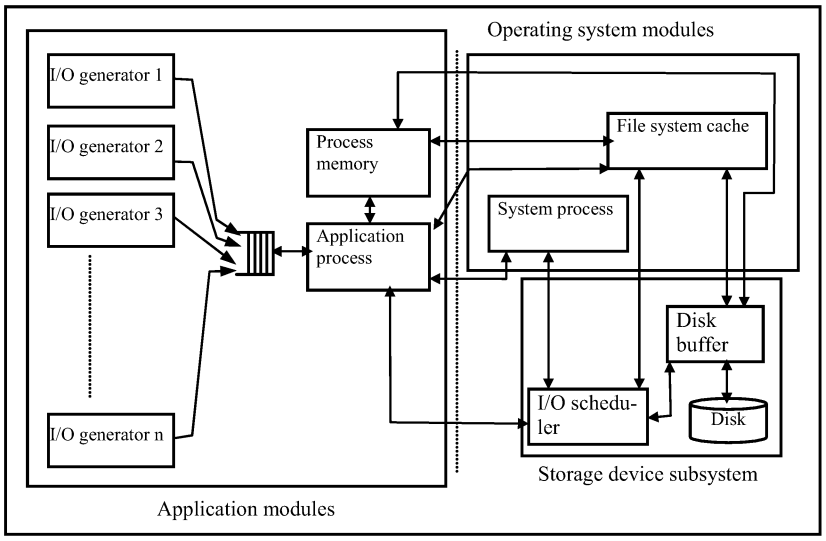


Figure 2: Architecture of the Windows I/O System Simulator

THE SIMULATOR'S KERNEL

When performing I/O operations on many configurations of Windows systems using many different disk drives (different manufacturers), we noticed that for a given workload there are always two important elements that control the performance pattern (in addition to the physical configuration of the machines):

The file system cache algorithms

For each access mode corresponds a prefetching algorithm except for the "no buffer" mode. We will try in the following sections to give a brief summary of what does the file system cache manager when receiving read/write requests:

For read operations:
a) The sequential mode: for this mode all the requests are divided (or concatenated) into 64KB requests. The file system cache prefetches the data block (64KB) sequentially, after loading the block Bn, the system prefetches automatically the block Bn+1.
b) The normal mode read: the prefetching algorithm in this mode is a bit more complicated than the latter. Two processes send requests: the application process and the system process. When we launch the first request, the system process begins loading the data requested by the application. Once the first request is satisfied (whatever the size), the system process jumps ahead one request size and begins to prefetch data. For a 256KB request (B1, B2, B3, B4 of 64KB) for example, the system loads the first four blocks (of 64KB) and goes prefetching the B9 block while the application is requesting for the B5 one.

For write operations:
a) Sequential and normal mode write: Two different behaviors were observed for these modes depending on the *request size*: first, the application process writes data only to the file system cache and never to the disk while the system process flushes that data *progressively*. In the second one, the application process writes data in both the file system cache *and* disk and the system process flushes *periodically* the data previously written on cache to the disk (once the dedicated memory workspace limit is reached):
- *Data flushed by the system process*: only the system process flushes the data to the disk in case of request sizes of 128KB, 256KB or ≤96KB. The flushing is done by different number of blocks of 64KB, in parallel with the copy of data by the application process to the file system cache.
- *Data flushed by the system and the application processes*: for a given data block, the application process writes periodically a part of the data to the disk and the remainder to the file system cache. This remainder is later on flushed by the system process on the disk subsystem (once the workspace is filled). The driver copies the blocks that the application process sends on the disk and leaves blanks where the data copied on the file system cache should be placed.

For instance, considering a 320KB request, the period is 4, during the first request, 3 blocks of 64KB are written on the disk and 3 blocks on the file system cache, for the second request, 4 blocks are written on the cache and 2 on the disk, the third request 5 blocks are sent to the cache and 1 to the disk and finally for the 4th request, all the blocks are written to the disk.
From the performed measures and after some analysis, we have inferred an algorithm to calculate the exact periodicity of the response times and the one used by the system to decide whether a block of data should be sent to the disk or to the cache and what is the size and the position of that block. The periodicities depend highly on the request sizes.
b) Write through mode: when a write request is issued, data are first copied into the file system cache (by 64KB blocks), they are then written to the disk cache and finally to the disk. Before issuing the next request, the system region corresponding to this file is modified.

All the algorithms described in this section were found to be OS dependent meaning that they *do not* depend on the used file system (FAT32 and NTFS) and are the same for the different Windows 2000 and Windows XP versions. One important point is that the disk behavior depends highly on the request sequencing provided by the OS. All of them were implemented in the simulator.

The disk cache algorithms

The disk cache generally performs prefetching operations. When reading one block of data (B1), if there are no requests waiting to be serviced, the disk loads a number of sectors contiguous to B1 (to fill the cache segment). These prefetching parameters depend on the disk configuration and may differ from a disk to another.
But if there are any requests waiting to be serviced, this prefetching will generally not happen. Instead, when the read/write head moves from one position to another one, it can load some data that are between the two. This depends on the distance between the two blocks on the disk and on their geometrical positions.
All these prefetchings, lazy write and write through algorithms are disk dependent and their behaviors depend on the sent request sequence (from the file system cache). Many of these algorithms were implemented on the simulator.

INPUTS / OUTPUTS OF THE SIMULATOR

The inputs of the simulator are either provided by the manufacturer (some of the disk characteristics for example) or can be extracted by the developed tool regrouping a set of tests and measures to identify the required parameters with a developed methodology. We have two main inputs for the simulator:

The workload generator

For each simulation we can define a different number of generators. Each generator is mainly characterized by its number of requests, request types and sizes, access modes, inter arrival times and addresses of data to access.

The simulated architecture parameters

To launch a simulation we need to have the parameters concerning the architecture of the disk, the disk cache characteristics, the interface throughput, memory to memory throughput, process workspace size, etc. To get all the re-

quired information, we developed a "push button" storage parameter extraction tool. This tool mainly identifies the memory to memory transfer rate, the cache disk to memory transfer rate, the track sizes, the track to track seek times, and the characteristics of the disk prefetching algorithms.

Table 1 : The Configurations of the Tested Machines for Validation

AMD athlon 1.41GHz	128MB RAM				
Maxtor 6L040J2	**Disk Cache**	**Capacity**	**Seek**	**Transfer rates**	
Interface	2MB	40GB	track to track 0.8ms	**Internal (max)**	**External (max)**
ATA 133	**Track sizes**	**Rotational speed**	average 8.5ms	54 MB/sec	133 MB/sec
	240ko / 441ko	7200 rpm	Full Stroke 17.2ms		
Windows XP family edition 2002				File system: NTFS	

Intel Pentium3 500MH	192MB RAM				
Toshiba MK6012MAP	**Disk Cache**	**Capacity**	**Seek**	**Transfer rates**	
Interface	1MB	6GB	track to track 3ms	**Internal (max)**	**External (max)**
ATA 4	**Track sizes**	**Rotational speed**	average 13ms	24 MB/sec	66.7 MB/sec
	unknown	4200 rpm	Full Stroke 24ms		
Windows2000 5.0.2195 service pack 2				File system: FAT32 & NTFS	

Intel Pentium4 1,8GHz 256MB RDRAM					
Fujitsu MAN 3184MP	**Disk Cache**	**Capacity**	**Seek**	Transfer rates	
Interface	8MB	18,4 GB	track to track 0,4 / 0,6m	**Internal (max)**	**External (max)**
Ultra 160 SCSI	**Track sizes**	**Rotational speed**	average 4,5 / 5ms	84 MB/sec	160 MB/sec
	221KB / 377KB	10025 rpm	Full Stroke 11 / 12ms		
Windows2000 Professional 5.00.2195 Service Pack 3				File system FAT32 & NTFS	

The outputs of the simulator are the performance of the system in terms of transfer rates and single response times of each generated request at each step in the simulation. The simulation can be completely viewed thanks to the OMNET++ allowing a graphical visualization of the simulation. The state of each module can be checked during the simulation: queues, service times, received messages, etc.

RESULTS AND VALIDATIONS

We evaluated the accuracy of our simulator by comparing its results with real measures realized on micro benchmarks in which we specify the complete I/O strategy (access modes, request sequence, request size, file size and position, etc). This allows us to guarantee the accuracy and reliability of the measures to test the simulator. We have done synchronous (synchronous accesses represent 69-91% of the I/O's in PC workloads (Hsu and Smith 2004)) sequential accesses (reads/writes) on 500MB files with request sizes varying between 64KB and 512KB. We measured the response times of every request for all the access modes and compared them with the simulator's results.

We considered three different configurations (see Table 1) using different versions of OS, two of the tested machines contain both an NTFS and a FAT32 partitions that show the same results. We also chose to test a notebook computer to see its behavior (machine configuration 2).

The simulator was set up for each configuration according to data provided by the manufacturer (10% of the parameters) and especially according to the parameters we have got with the developed extraction tool.

The generators of the simulator were set to simulate synchronous accesses with zero inter arrival times and the region accessed on the disk were measured with the parameter extraction tool (track sizes).

The simulator results for the tested modes and for a set of request sizes are given in Figure 3 and Figure 4. Figure 3 depicts some result samples obtained from the simulated reading operation compared to those actually measured for the three configurations on the "no buffer" mode and normal mode for some request sizes. We noticed a nearly perfect matching between the simulated and measured response times (less than 3% error margin). We observe that the periodicities are always the same.

The write operations (Figure 4) show nearly a perfect matching between the measured and the simulated response times. The small amount of points that does not correspond to the simulated values is due to the presence of spare sectors and cylinders, but even with those errors (that occur rarely) the global transfer rate is maintained within the 3% error margin. For lack of space reasons, only the results of the first configuration are shown.

We also show the validation of the multi file read operation which is equivalent to a multi generator read operation on 2 files with 64KB request sizes using the no buffer mode. Figure 6 shows a periodic (~200 requests) response time variation due to changing angular distances between the requested data blocks on the disk in spite of their equal sizes. Since the recent disks are multi-zoned (Van Meter 1997), if two files are on two different zones, and are accessed in a perfectly interleaved way, the angular distance between blocks changes for each pair of requests and the positioning time increases up to its maximum value and then decreases down to its minimum value.

A comparison of global transfer rates obtained from the measurements and simulations is shown in Figure 5. It shows an excellent matching between the simulated and measured results (about 3% error margin in the worst case).

RELATED WORK

While storage system simulators (Bucy et al.) (Wilkes 1996) and system extraction tools (Schindler and ganger 1999) (limited to SCSI systems) do exist on Linux/Unix systems, they are still inexistent on Windows OS which is dominant in the industrial world and relatively absent in academic research. Their architecture and especially file system cache behavior (Shriver and Small 1999) under both systems is very different. The used prefetching, write back and write through algorithm implementation are not the same. On the other hand, on the tested Windows systems (Boukhobza 2004), the behavior of that entity is the same on all versions of Windows XP and all versions of Windows 2000, whatever the file system (NTFS or FAT32). This allowed us to create a Windows I/O simulator based upon the Windows file system cache behavior that was investigated by test.

Various works concerning global I/O performance measuring of the Windows storage system have been achieved (Riedel et al. 1998) (Chung et al. 2000). Riedel *et al.* explores the storage system performance for the WindowsNT4 operating system. Multiple disk performance was also investigated. Different architectures and technologies were presented and the relative performance of the different storage components measured. The work done by Chung *et al.* is a continuation of (Riedel et al. 1998) where different storage system performance was measured under Win-

dows2000. Comparisons of the measured global performance have been made between different operating systems (WindowsNT4 and Windows2000) and different interfacing technologies. Although this work spans a variety of storage architectures, it is limited to a global presentation of the exhaustive obtained measures and lacks a precise performance impact for the involved parameters.

CONCLUSIONS AND PERSPECTIVES

We exposed in this paper an I/O system simulator for Windows OS. It simulates very accurately all the storage system behavior from the I/O generator till the mechanical movement of the disk arm. It covers all of the different cache algorithms for the Windows file system cache, disk cache, in addition to the mechanical characteristics of disks including track size, seek times, zoning, head switching time, track and cylinder skew, etc. The I/O bus was also simulated: sharing of the bus, the real throughput of the bus, the I/O scheduler, the memory copy speed, the system bus, etc.
To launch a simulator on a given architecture we have to define a workload and the specification of the architecture to measure (see Figure 7). The first one is relatively easy to define, it is supplied by the application designer who tests a given modeled application I/O strategy. But it is generally harder to get the storage system parameter specifications. We developed for that sake a parameter extraction tool based on "black box" identification that makes it very easy for the simulator user to get the storage system parameters.
The results of the simulator provide to the developer a precise performance estimation of the selected IO strategy on the tested storage system in terms of transfer rates, average, min and max response times, etc.
Another advantage of the simulator is that it gives the possibility to investigate future configurations (or extension of existing architectures).
The obtained results show a very good matching between the measured and simulated performance. Even though the shape of the curves and periodicities (response times) are very different between the different configurations, the simulator is able to cope with those differences very accurately. We even compared some of the extracted parameters with the manufacturer's data (Fujitsu 2001) (when provided) and we obtained the same values.
The methodology and tools developed are being used to test and simulate other storage systems and applications and we are validating it on over ten different configurations. The ability to plug real traces is being under test. The traces are for the moment generated with the help of the *Filemon* tool developed by *Sysinternals* (www.sysinternals.com). Furthermore, an asynchronous request support extension of the workload generator is under test.
The simulator is available on www.prism.uvsq.fr/users/~jboukh.

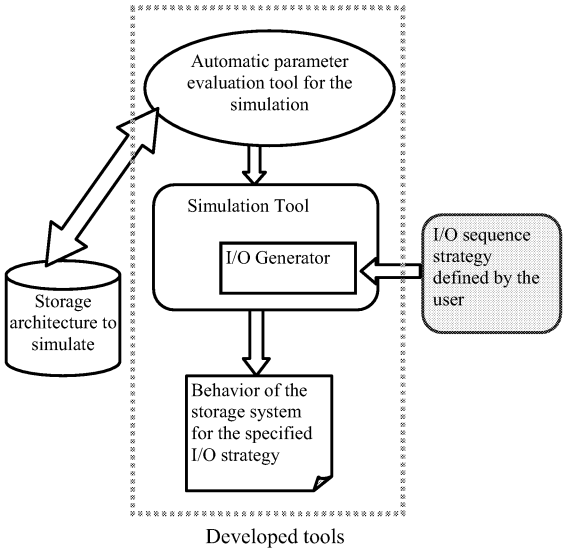


Figure 7: Implemented Tools Including the Measurement Tools and the Simulator

ACKNOWLEDGEMENT

These studies are partially financed by France Telecom R&D. Special thanks to Hamza Aissat who collaborated in the development of the simulator.

REFERENCES

Boukhobza, J. 2004. "Transfer rate variations of file accesses on Windows". Technical report *#2004/56* PRiSM laboratory, Versailles University, Fr. (Apr).

Bucy. J. and G. Ganger. 2003. "The disksim simulation environment version 3.0 reference manual". Technical report CMU-CS-03-102, Carnegie Mellon University, Pittsburgh. (Jan).

Chan, S.G. and F.A.Tobagi. 2003. "Modeling and dimensioning hierarchical storage systems for low-delay video services". *IEEE Transactions On Computers*, Vol52, No. 7. (Jul).

Chung, L. ; J.Gray; B.Worthington ; and R.Horst. 2000. "Windows2000 disk I/O performance". Technical report MS-TR-2000-55. Microsoft Research Advanced Technology Division. (Jun).

Fujitsu. 2001. *Product/maintenance manual, MAN3184 MC/MP series disk drives*. C141-E128-01E, Fujitsu limited.

Hsu, W.W. and A. J. Smith. 2004. "The performance impact of I/O optimizations and disk improvements". *IBM J Res. & Dev.*, Vol. 48, No. 2. (Mar).

Oney, W. 1999. *Programming the Windows Driver model.* Redmond Washington, Microsoft Press.

Riedel E. ; C.VanIngen ; and J.Gray . 1998. "A performance study of sequential I/O on WindowsNT4". *Proceedings 2nd USENIX WindowsNT Symposium*, Seattle, WA. (Aug).

Schindler, J. and G.Ganger. 1999. "Automated disk drive characterization", CMU-CS-99-176. Carnegie Mellon University, Pittsburgh. (Dec).

Shriver, E. and C. Small. 1999. "Why does file system Prefetching work?". *Proceedings of USENIX 1999 Annual Technical conference*, Monterey California USA. (Jun).

Solomon, D.A. and M.E.Russinovich. 2000. *Inside Windows2000,* Redmond, Washington, Microsoft Press.

Thekkath, C.A. ; J.Wilkes ; and E.D.Lazowska. 1994. "Techniques for file system simulation", *Software practice and experience,* Vol, 24(11), 981-999. (Nov).

Van Meter R. 1997. "Observing the effects of Multi-Zone disks". *Proceedings of USENIX 1997 Annual Technical conference*, Anaheim, CA, (Jan).

Varga, A. 2003. *OMNET++ discrete event simulation system, user manual*, www.omnetpp.org. (Feb).

Wilkes, J. 1996. "The Pantheon storage system simulator", HPL_SSP_95_14. Computer Systems Laboratory Hewlett-Packard Laboratories, Palo Alto, CA. (May).

Worthington, B. ; G.Ganger ; and Y.Patt. 1994. "Scheduling algorithms for modern disk drives". *Proceedings of ACM Sigmetrics*. (May).

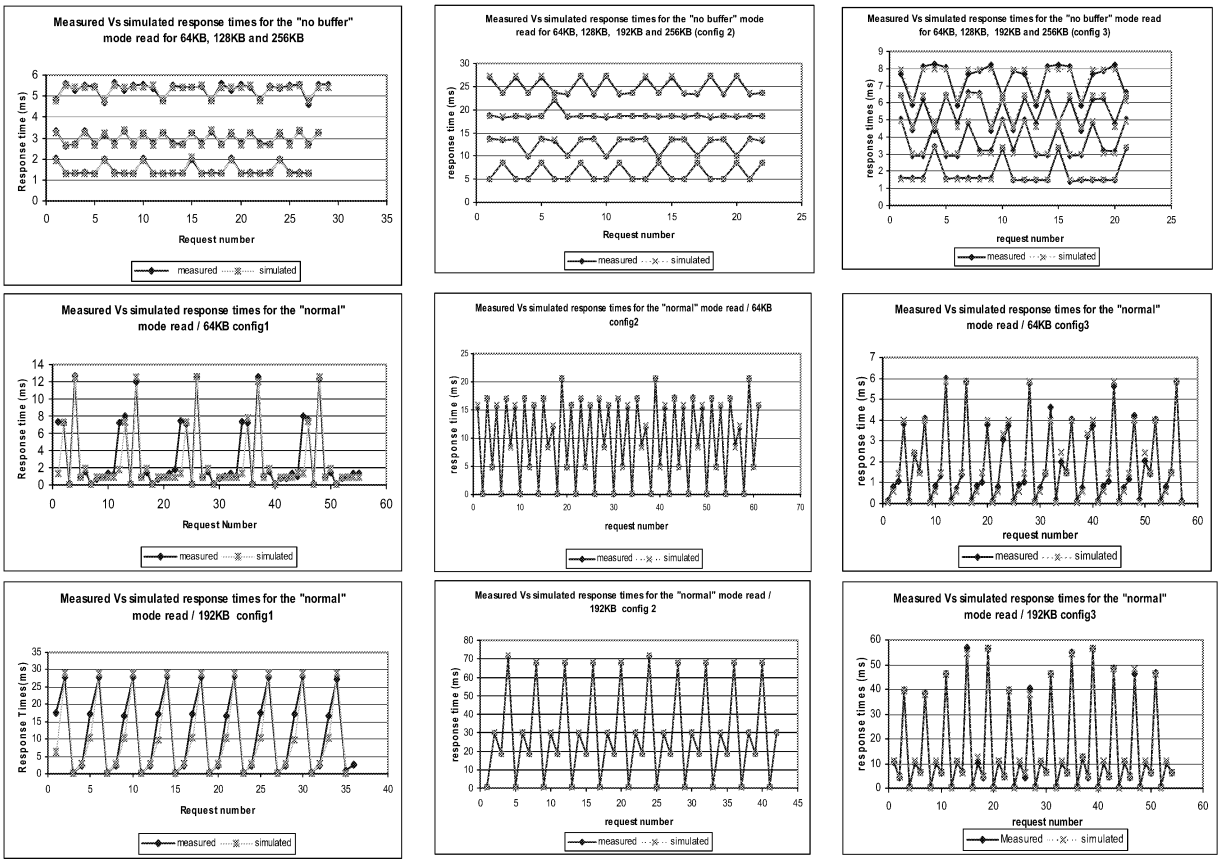


Figure 3: Measured and Simulated response times for the read operation for different request sizes for the “no buffer” and normal modes. All the results could not be shown, for the sequential mode for example, the shape of the response time curve is similar to the one of the “no buffer” mode, the difference is that for the sequential mode, it is the system process that satisfies the requests. For the normal mode, only two figures are shown to give an idea of the obtained performances.

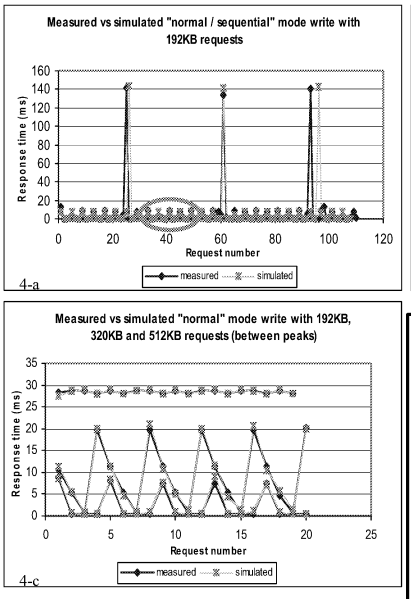


Figure 4: Measured and Simulated response times for the “normal” 5-a and 5-b (196KB, 320KB and 512KB) and “write through” (64KB,256KB) mode (5-c) operation for config 1

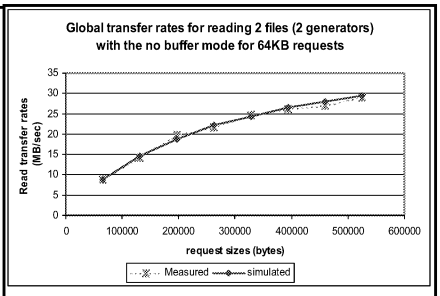


Figure5: Measured and Simulated transfer rates for the “no buffer” mode for different request sizes on two different files (2 generators) for config 1

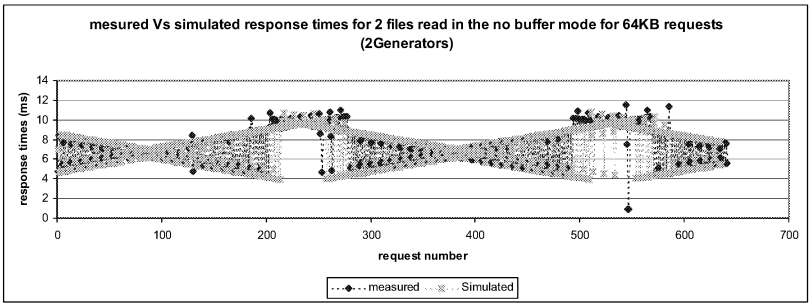


Figure 6: Measured and Simulated response times for the “no buffer” mode 64KB read operation on two files (simulated by 2 generators) for config 1

DACSIM: A DATA ACQUISITION CARD SIMULATOR

Antonio Martí
Escuela Técnica Superior de Informática Aplicada
Departamento de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia
Camino de Vera s/n, 46022 Valencia
Spain
E-mail: amarti@disca.upv.es

Salvador Coll
Departamento de Ingeniería Electrónica
Universidad Politécnica de Valencia
Camino de Vera s/n, 46022 Valencia
Spain
E-mail: scoll@eln.upv.es

Marina Alonso, Juan José Serrano
Escuela Técnica Superior de Informática Aplicada
Departamento de Informática de Sistemas y Computadores
Universidad Politécnica de Valencia
Camino de Vera s/n, 46022 Valencia
Spain
E-mail:{malonso, jserrano}@disca.upv.es

ABSTRACT

Currently data acquisition cards are being widely used by process control engineers. In this paper a data acquisition card simulator, DACSIM, developed with teaching purposes is presented. In addition a virtual laboratory has been developed, in order to provide the simulator with signals equivalent to those generated by real processes and their related instrumentation.

DACSIM is composed of four building blocks: the virtual device controller (vxd), that provides an interface between the user application and the data acquisition card; the data acquisition card simulator, that reproduces the behavior of a real data acquisition card; the virtual laboratory, that provides virtual instrumentation and a set of industrial processes simulator boards, that reproduce real control systems.

One important contribution of DACSIM is the fact that its operation is completely transparent to the application, since the code do not need any modification with respect to the one that runs on a real data acquisition card.

INTRODUCTION

The main goal of DACSIM is to allow the use of any personal computer, by simply installing the appropriate software and without acquiring or installing any hardware device. To achieve this goal, a detailed knowledge of the signal generation and acquisition process performed during a normal program execution is required. Figure 1 shows this process in detail.

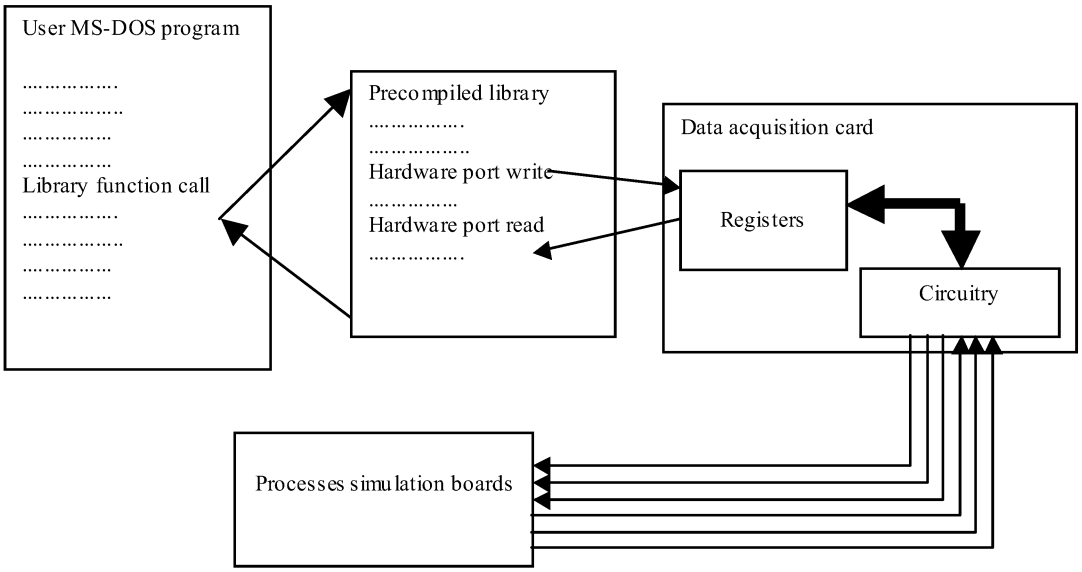


Figure 1. Communication between the user application and the processes simulation boards.

The call to a library function by the user program involves the execution of a sequence of instructions that access the computer hardware ports. The data acquisition card registers are mapped on this input/output ports.

Depending on the called function (generate an analog output voltage, read a digital input, etc.) particular data values are written into the appropriate registers. Although this information is hardware dependent, it is hidden to the user by the programming libraries. Once the command and the data have been written into the registers, the data acquisition card performs the requested operation, either generating an output control signal, or acquiring a feedback signal. The user can supervise the behavior of the running control algorithm and analyze the results by means of the process simulation boards.

It is required for DACSIM to communicate with an MS-DOS task (the user application) through the hardware input/output ports. This is done in three steps:

- First the values written by the library functions into the card registers are intercepted.
- Second, these values are interpreted as data and control messages.
- Third, the real data acquisition card operation is simulated by returning the appropriate values in the registers according to the requested operation.

In addition, the simulator must include a graphic interface to allow a simple way for the user to read the data acquisition card outputs and write to the inputs, in a similar way that it would be done in the laboratory.

To achieve the above goals the solution has been based on three software blocks:

- Windows virtual device driver, whose task is perform the communication between the MS-DOS application and the data acquisition card simulator using the input/output ports.
- Data acquisition card simulator, that receives the data sent through the ports from the virtual device driver, proceses them and send the information to the virtual laboratory. And viceversa.
- Virtual laboratory, that includes a graphic interface to the process simulation boards, which the user will be able to interact with. The virtual laboratory may be substituted by a virtual process.
- Virtual processes, that simulate complete processes.

Figure 2 shows the communication among the basic software components of the proposed simulator. Each building block is described in the following sections.

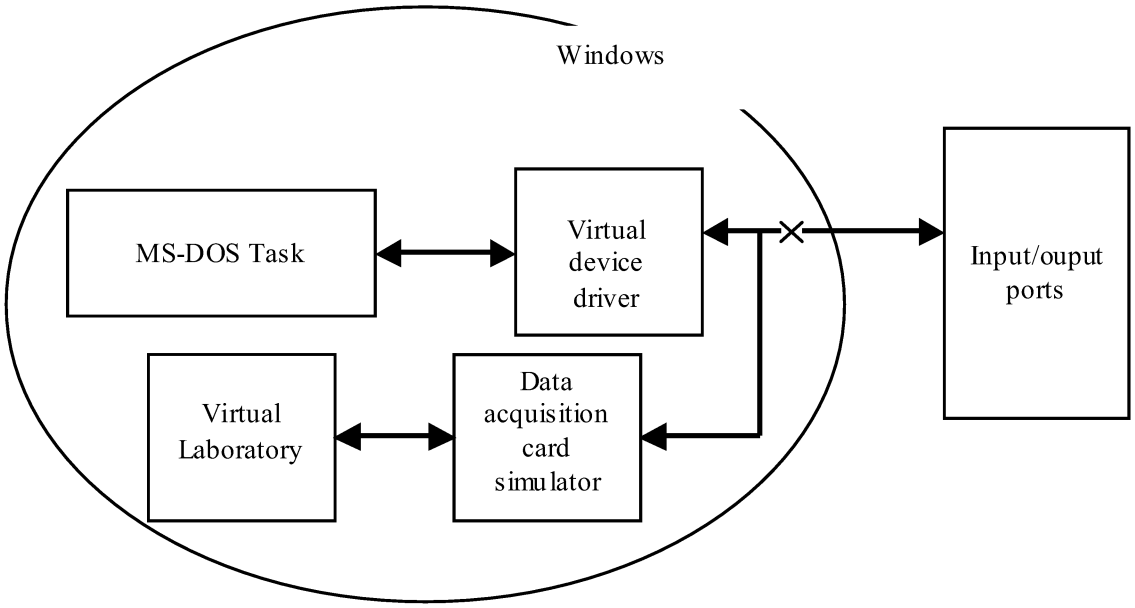


Figure 2. MS-DOS – virtual laboratory communication

Virtual Device Driver

One of the main goals and difficulties to develop DACSIM has been to make it work with no user application or libraries modifications. This restriction has required the development of a communication technique between two tasks: the user program and the simulator. In Windows, the communication between win32 tasks is solved by message passing, but this is not supported for MS-DOS tasks. It might be possible to do that by means of some system call, but this would require user code modifications. Another alternative could be the use of a C compiler for Win32

applications. Nevertheless, this would make the program development even harder, since the user should develop not only the control algorithm but also the graphic interface.

In order for this communication to be transparent, it has been necessary to exploit the hardware resources protection offered by Pentium processors, managed by Windows through exceptions. In Windows, if an input/output port is not reserved, the operating system allows access to this hardware resource. However, if some task has requested the exclusive use of that resource, the operating system generates an exeception against any access that will have to be treated by the corresponding rutine.

The developed virtual device driver requests to the operating system the reservation of the port adresses corresponding to the data acquisition card mapping. From this point, any access to those addresses will generate an exception. This will execute the virtual device driver that performed the port reservation. If it has been a write access, the operating system provides the rutine with the accessed port and the written value. If the access has been a read access, the operating system provides the device driver with the accessed port and requests it the value to be returned to the task that performed the access (Figure 3). In our proposal, the virtual device driver behaves as a bridge program between the MS-DOS task (user program) and the simulator. Its only function is capturing the accesses to input/output ports and redirecting them to the simulator.

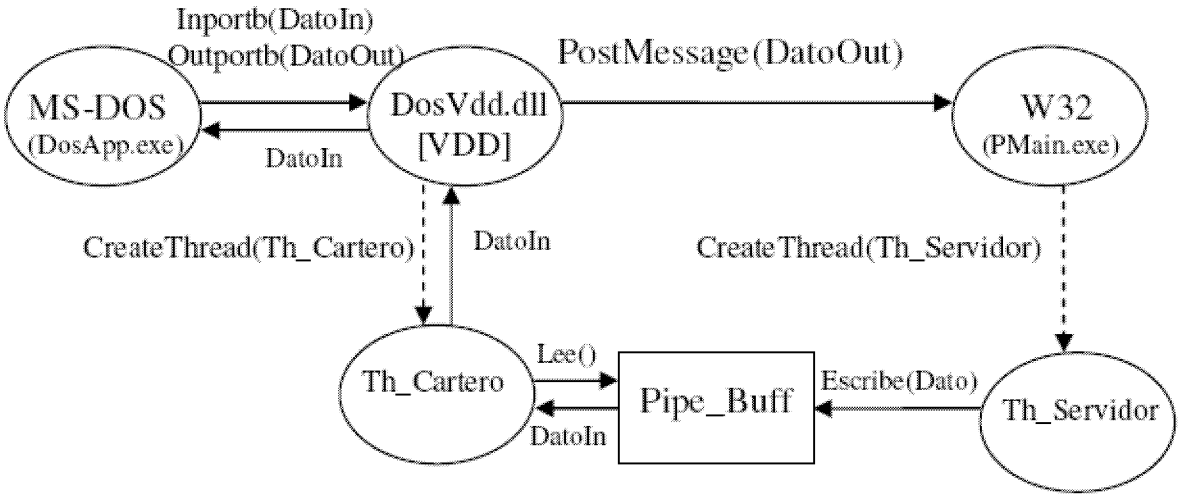


Figure 3. Communication details among the MS-DOS application and the device driver.

Data Acquisition Card Simulator

The basic operation of the simulator is described in this section.

When the virtual device driver writes a piece of data in a register that modifies some of the card analog outputs, the simulator converts this digital word into its corresponding voltage value, and sends it, together with the output channel, to the virtual laboratory.

When the virtual driver reads the contents of any of the registers containing the state of the inputs, the simulator requests the virtual laboratory the particular data and converts it into its binary form.

Additionally, the simulator is capable of generating interrupts if requested by the user program.

In order to provide the user with information regarding the data acquisition card state, or with the capability of performing tests without writing as program, the simulator has a window (Figure 4) that shows the state of the card registers and their evolution while they are being accessed. In addition, the user can modify the content of any register using this window.

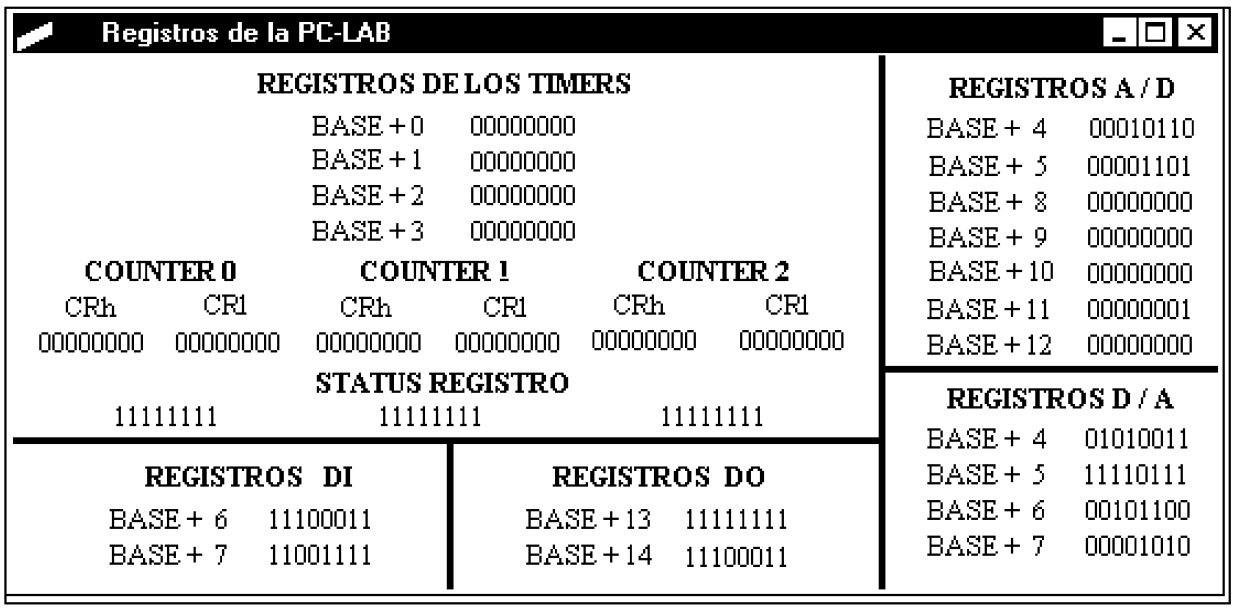


Figure 4. Data acquisition card simulator interface window.

Virtual Laboratory

During the conventional laboratory sessions, to verify the correct operation of a user control program, the student is provided with two process simulation boards, connected to the data acquisition card, that display the program generated outputs and allow modifying the inputs. In particular, this boards provide:

- 8 LEDs connected to 8 digital outputs of the card;
- 8 microswitches connected to 8 digital inputs;
- an analog voltmeter and a lamp connected to the analog outputs; and
- a potentiometer and an LDR connected to two analog outputs.

These devices do not use the complete set of available input/outputs, but are enough to develop the exercises developed during the course (if used for teaching purposes) or to illustrate the behavior of many real applications (if used for control algorithms development).

Although the goal of the virtual laboratory is to reproduce these devices, the zero cost of including additional devices in the virtual laboratory has allowed the increase of the virtual instruments as compared to the real ones available in the laboratories. For this reason the virtual laboratory, whose layout is depicted in Figure 5, includes:

- 16 LEDs connected to the 16 digital outputs of the card;
- 16 microswitches, connected to the 16 digital inputs;
- 2 analog voltmeters, connected to the analog output channels (one of them is also used by a lamp); and,
- 16 potentiometers, connected to the 16 analog input channels (one of them is also used by the LDR, which is excited by the lamp and can be covered by the umbrella).

As mentioned above, any modification in the outputs by the user program will be shown on this window. Simultaneously, any change performed by the user in the inputs is transferred to the data acquisition card simulator and, hence, to the user program.

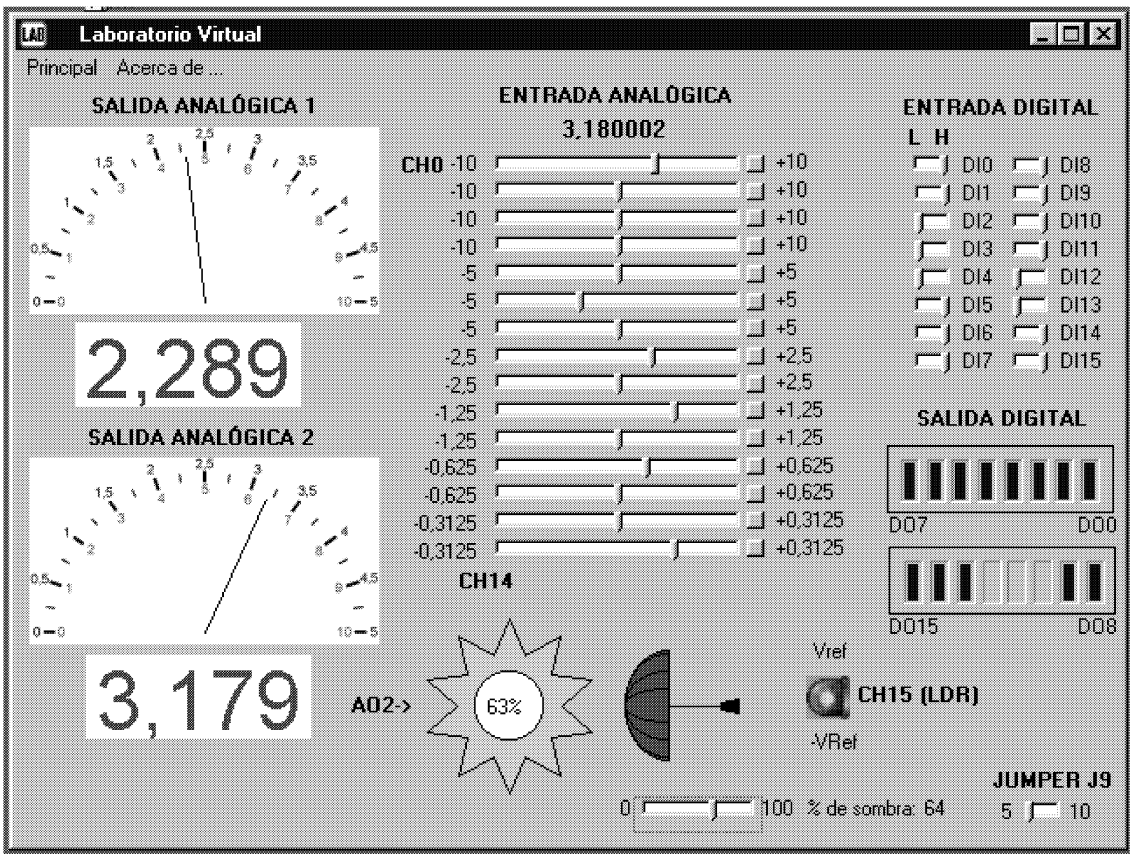


Figure 5. Virtual laboratory.

Virtual Processes

As an alternative to the virtual laboratory, a set of complete processes simulators have been developed. In this way, the student can substitute the general purpose virtual laboratoty block with an application-specific simulator. This technique can be used to simulate more complex applications without the need for any hardware.

CONCLUSIONS

In this work we have presented a data acquisition card simulator, DACSIM, that makes it posible to develop real applications without specific hardware (including the data acquisition card). One of the main contributions of DACSIM is its completely transparent behavior. That means, that the user do not need to modify the application to run either on the real card or on the virtual one.

In addition, the development of a virtual laboratory, connected to the simulator input/outputs, that reproduces the laboratory instrumentation, allows an efficient and comfortable user work.

The presented versión of the simulator has used as starting point MS-DOS based applications. The reason for that is the fact that, the main objective of the subject for which this tool has been developed is the programming of control algorithms using data acquisition cards. Thus, the programming of graphical interfaces is avoided. However, versions for different application sources are under development.

We believe in the need for the utilization of real devices by engineering students. Nevertheless, DACSIM apart from being a powerful teaching tool, introduces the student in the use of simulation as a preliminary step to the development of prototype or even real applications.

FUTURE WORK

The future work is planned in three lines. First, developing new simulators coping with new models of commercial data acquisition cards. At the moment the simulator has been developed for a ISA card. This type of cards is being broadly substituted by PCI cards that present several differences concerning their structure and operation, as well as the way the virtual device driver must access the simulator.

Second, the simulator is designed for user applications developed in MS-DOS. The use of visual applications, as Borland C++ Builder, Jbuilder or Visual C++, for data acquisition cards programming is becoming increasingly common. The communication between the user application and the card simulator may be, in this case, simpler by using the messages of the operating system. However, this

forces the utilization of a library different to the one provided by the card manufacturer, forcing the user to distinguish when it has the physical card and when it only has the simulator. This work must be carefully planned.

Finally, new process models are being developed, coping with the problems the student might find in other subjects.

REFERENCES

[1] Leblanc, Gérard. "Borland C++ Builder" Barcelona. Ediciones Gestión 2000, 1997. ISBN 8480881844.

[2] Richter, Jeffrey. "Programación avanzada en Windows" Madrid. Mc Graw Hill, 1997. ISBN 8448111605

[3] Thielen, David; Woodruff, Bryan. "Writing Windows: Virtual Device Drivers" Addison-Wesley, 1994. ISBN 020148921X

[4] "ACL-8112pg Advanced Multifunction Data Acquisition Card" ADLink Technology, 1997.

[5] "Jeremy Gordon's Win32 + assembler source page" http://www.GoDevTool.com

[6] Durá Alepuz, Enrique; Martí Campoy, Antonio (director) "Laboratorio virtual de informática industrial" Proyecto final de carrera de la ETSII de la Universidad Politécnica de Valencia. 2000

[7] Martí Campoy, Antonio; Ors Carot, Rafael; Pérez Jiménez, Alberto. "Laboratorio de Informatización Industrial" Editorial de la Universidad Politécnica de Valencia. (cd-rom), 2001. Ref. UPV 2001-870

[8] Perles, Ángel; Martínez, Juan M.; Hassan, Houcine; Albadalejo, José; Domínguez, Carlos. "El simulador SimSeny en la innovación docente de la asignatura Informática Industrial" VII Jornadas de Enseñanza Universitaria de la Informática. Palma de Mallorca, Julio de 2001.

JARA 2i – A MODELICA LIBRARY FOR INTERACTIVE SIMULATION OF PHYSICAL-CHEMICAL PROCESSES

Carla Martin, Alfonso Urquia and Sebastian Dormido
Dept. Informática y Automática, E.T.S. Ingeniería Informática, U.N.E.D.
Juan del Rosal 16, 28040 Madrid, Spain
E-mail: {carla, aurquia, sdormido}@dia.uned.es

KEYWORDS
Interactive simulation, object-oriented, continuous simulation, education, automatic control.

ABSTRACT

Virtual laboratories, supporting interactive dynamic simulations, can be easily implemented by combining three software tools: Ejs, Matlab/Simulink and Dymola (with Modelica language). This approach has been successfully applied to set up virtual labs, with the intent to be used in education of the control of chemical processes. To achieve this goal, the following tasks have been completed: (1) the object-oriented design and programming of JARA: a Modelica library of dynamic hybrid models of some fundamental physical-chemical principles; (2) the description of the JARA physical models in a way suitable for interactive simulation; and finally (3) the implementation of the virtual labs. The virtual-lab view has been developed using Ejs, the physical models using JARA library and the controllers using Simulink block diagrams. The design of JARA 2i is discussed in this manuscript. In addition, its application to the modeling of virtual labs is illustrated by means of a case study (the control of an industrial boiler).

INTRODUCTION

Virtual laboratories supporting interactive simulations are effective pedagogical resources. During the interactive simulation run, the students can change the value of the model inputs, parameters and state variables, perceiving instantly how these changes affect to the model dynamic. The students' understanding is enhanced by the interaction possibilities and this promotes their motivation to study the subject.

A methodology well suited for virtual-labs implementation is discussed in (Martin et al. 2004). It is based on combining the use of Ejs, Matlab/Simulink and Modelica/Dymola. This approach takes advantage of the best features of each tool. Ejs' capability for building interactive user-interfaces composed of graphical elements, whose properties are linked to the model variables (Esquembre 2004; http://fem.um.es/Ejs/). Matlab/Simulink's capability for modeling automatic control systems and for model analysis (http://www.mathworks.com/). Modelica's capability for physical modeling (http://www.modelica.org), and finally Dymola's capability for simulating hybrid-DAE models (Dynasim 2002).

Ejs is a freeware, open source, Java-based tool intended to program web-based virtual labs. It is easy to install and use (http://fem.um.es/Ejs/). Ejs guides the user during the definition of the virtual lab, and it automatically performs all the tasks required to generate the virtual-lab code: a Java application or an applet.

Ejs is based on a simplification of the "model-view-control" paradigm. Ejs structures the virtual-lab definition in the following three parts:

1. *Introduction*: html pages including educational content related with the virtual-lab topic.
2. *Model*: dynamic model whose interactive simulation is the virtual-lab basis.
3. *View*: user-to-model interface. It is intended to provide a visual representation of the model dynamic behavior and to facilitate the user's interactive actions on the model.

Ejs includes a set of ready-to-use visual elements, that allows easy creation of the virtual-lab view. The graphical properties of the Ejs' view elements can be linked to the model variables, producing a bi-directional flow of information between the view and the model. Any change of a model variable value is automatically displayed by the view. Reciprocally, any user interaction with the view automatically modifies the value of the corresponding model variable.

Ejs gives the user a procedure to define the model, and provides set of built-in ODE solvers to simulate it. Ejs virtual labs can run stand-alone or in conjunction with Matlab/Simulink. In this last case, the model can be partially or completely described using Matlab code and Simulink block diagrams (http://fem.um.es/Ejs/). Ejs synchronizes the numerical solution of the complete model. The model part described using Matlab/Simulink is solved using the Matlab workspace and the Simulink's numerical algorithms.

On the other hand, Modelica language and Simulink block diagrams can be jointly used for model definition.

Dymola 5.0 interface to Matlab 5.3 / Simulink 3.0 can be found in Simulink's library browser: DymolaBlock block (Dynasim 2002). This block is an interface to the C-code generated by Dymola for the Modelica model. This C-code contains the numerical algorithms required to simulate the Modelica model, which is formulated in terms of differential-algebraic equations (DAE) and discrete-time equations. DymolaBlock block can be connected to the Simulink blocks and also to other DymolaBlock blocks.

Contributions of this Paper

This software combination approach has been successfully used to program a set of virtual labs for chemical process control. To achieve this goal, the following three tasks have been completed:

1. Object-oriented design of JARA (Urquia and Dormido 2003; Urquia 2000). Briefly, JARA is a set of libraries of dynamic hybrid models of some fundamental physical-chemical principles. The main application of JARA is the modeling of physical-chemical processes in the context of automatic control.
2. The description of JARA physical models in a way suitable for interactive simulation using Ejs. The modeling methodology proposed in (Martin et al. 2004) has been applied. The JARA library version, that is written in Modelica language and intended for interactive simulation using Ejs, is JARA 2i (release July 2004).
3. Implementation of the Ejs virtual labs. The physical models of the chemical plants are composed using JARA 2i, and they are embedded within Simulink's DymolaBlock blocks. The controllers are described using Simulink's block diagrams.

The design and implementation of JARA 2i is discussed in this manuscript, and its application to the development of Ejs virtual-labs is illustrated by means of a case study: the control of a boiler.

DESIGN OF JARA 2i

The methodology applied for designing JARA was proposed in (Urquia 2000; Urquia and Dormido 2003). The fundamentals of this methodology and its application to the JARA design are discussed next.

JARA's Design Methodology

JARA's design methodology is based on the concepts of the object-oriented modeling, fully exploiting the principles of abstraction and information encapsulation. It distinguishes between the role of the library designers/programmers and the role of the library users. One of the main goals of this methodology is to guarantee that the users are able to use the model libraries without having to know their internal details. In particular, the users should not be confronted with numerical problems. A key idea is the following: the identification and solution of the numerical difficulties associated to the library use should be carries out (as far as possible) by the library designers. To achieve these objectives, non-declarative techniques are recommended in the library design phase: causal analysis, explicit selection of tearing variables, decoupling via dummy dynamics, etc.

Another methodological key concept is the *library design rules*. They cover the numerical aspects of the design and use of the libraries, defining: (1) the complete set of requirements that each library's model has to fulfill; (2) how each library's model can be used and what models are interchangeable; and (3) how the libraries can be enlarged with new classes to be used in conjunction with the existing ones. The two pieces of information that the library designers have to provide to the library users are: (1) the hypothesis which each model of the library is based on; and (2) the library design rules.

The JARA design methodology is composed of the following five steps (Urquia and Dormido 2003):

1. *Phenomena and hypotheses*. The phenomena to be modeled and the general modeling hypotheses are defined. Once all the pairs of phenomenon to be modeled – modeling hypothesis, have been defined, then they can be grouped into libraries.
2. *Phenomena interaction: causal-connector classes*. A causal-connector class is defined by means of the following four pieces of information: (i) the variables composing the connector, classified into *across* and *through*; (ii) the allowed computational causalities of the connector class; (iii) the way of breaking the computational causality loops; and (iv) a graphic icon. All the classes of causal connectors should be gathered in a library.
3. *Connection rules of the causal connectors*: nature and number of the allowed connections among causal connectors. The library *design rules* are stated during the second and third steps of the design. The design rules are composed of the causal connector definition and their connection rules.
4. *Definition of the interface classes*. These interfaces should be only composed of an arbitrary number of causal connectors. The interface classes should be gathered in a library.
5. *Description of the phenomena*. The rules imposed by the interface's causal-connectors strongly condition the phenomena description. Techniques for facing these conditionings are discussed in (Urquia 2000; Urquia and Dormido 2003).

Fundamental Modeling Hypotheses of JARA

The usual way of enunciating the mass, energy and momentum balances is by means of the definition of a *control volume* (Bird, Stewart and Lightfoot 1975; Incropera and DeWitt 1996; Himmelblau and Bischoff 1992). The properties of the medium inside the control volume are considered time-dependent, but independent of the spatial coordinates. The only exception to this rule is the pressure inside the liquids. The control

volumes exchange mass and energy with their environment through certain *control planes*. The JARA control volumes and the control planes are considered macroscopic and fixed in the space. All the interactions among control volumes, and all the interactions of a control volume with itself (i.e., chemical reactions inside the control volume), are considered *transport phenomena* in JARA. This system decomposition into control volumes and transport phenomena suggests that:

1. The control-volume models should contain the equations describing the properties of the medium inside the control volume, as a function of the mass and energy transport through its control planes.
2. The transport-phenomena models should contain equations describing the flow of mass, energy and momentum through the control planes, as a function of the medium properties at these control planes.

Three types of control volumes have been modeled in JARA: (1) control volumes containing an ideal, homogeneous liquid mixture, composed of an arbitrary number of components; (2) control volume containing an ideal mixture of an arbitrary number of semi-perfect gases; and (3) control volume containing a homogeneous solid. The control volumes containing liquid or gaseous mixtures are considered open systems (i.e., they can exchange mass and heat with their environment), and chemical reactions can take place inside them. In both cases, the volume of the control volume is considered a time-dependent property. The control volume containing a solid is considered a closed system (i.e., it only exchanges energy, not mass, with its environment). The only modeled characteristic in solids is the heat conduction (for modeling the walls of reactors, pipes, etc.).

Two kinds of control planes are distinguished in JARA: mass-flow and heat-flow control planes. An arbitrary number of flows can flow through each control plane. The hypotheses made about the properties of the medium inside the control volume determine the nature and number of the control planes:

1. A solid control volume contains only one heat-flow control plane: (i) the solid properties are spatially homogeneous, so that all control planes are equivalent; and (ii) the solid control volume is a closed system, so it has no mass-flow control planes.
2. A gaseous control volume contains one heat-flow control plane and one mass-flow control plane: all the gaseous mixture properties are spatially homogeneous.
3. A liquid control volume has two mass-flow control planes and one heat-flow control plane: (i) as the liquid properties related with the heat-flow are spatially homogeneous, all the heat flow control-planes are equivalent; (ii) as the liquid pressure depends on the position, the simplest and most general control-plane selection is placing a control plane at the control-volume bottom and the other at the control-volume top. Any arbitrarily complex configuration can be modeled by decomposition into this kind of control volume (Urquia 2000).

The JARA models of transport phenomena can be divided into two main groups: (1) *mass transport* due to the pressure or concentration gradients, the gravitational acceleration, chemical reactions, liquid-vapor phase changes, etc.; and (2) *heat transport* due to the temperature gradient. The interface variables are grouped into connectors according to this criterion, so that they describe the transport of mass and heat independently.

A hypothesis related to the stirred-mixture approximation is to assume that the fluid going out from a control volume has the same properties than the fluid contained in it. In JARA, this approximation is applied to the calculus of: (1) the temperature and the composition of the fluid leaving a control volume by convection; and (2) the temperature of each mixture component leaving the control volume by diffusion. All the JARA models of transport phenomena make the flow direction reversible during the simulation run. As a consequence, the properties of the flow established between two control planes are calculated from the appropriate control plane at any time.

An important property associated to the transport phenomena is the *transport delay*. There are different ways of modeling delays in one-dimensional geometry systems (EPRI 1984). The way used in JARA is the "energy balance method" (Incropera and DeWitt 1996). It consists in dividing the flow path into multiple control volumes. Adjacent control-volumes are connected by the transport-phenomena models describing the heat and mass transport between them. As the number of control volumes increases, the solution gets closer to a transport delay (EPRI 1984).

Modeling for Interactive Simulation

The modeling methodology proposed in (Martin et al. 2004) has been applied to the JARA 2i programming. This methodology states how a Modelica model can be formulated to suit interactive simulation using Ejs. The obtained Modelica model fulfills the following two requirements:

1. In order to embed the Modelica model within a Simulink block, the computational causality of the Modelica model interface needs to be explicitly set.
2. As a result of the user interaction, the Modelica model needs to support instantaneous (discontinuous) changes in the state, and in the value of its input variables and parameters. In addition, several choices of the state variables can to be simultaneously supported, in order to provide alternative ways of describing the state changes.

The reader is referred to (Martin et al. 2004) for a complete discussion.

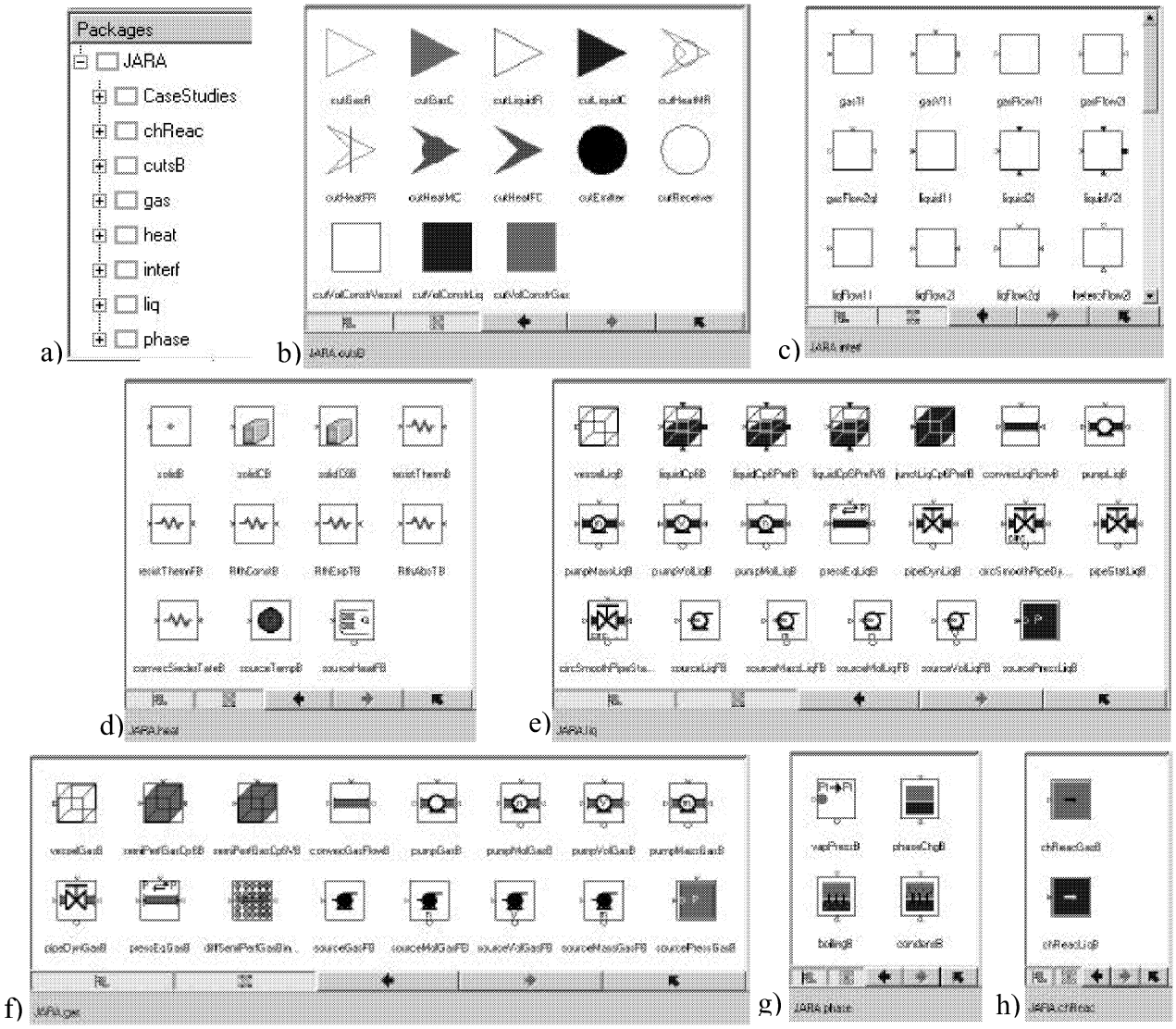


Figure 1: a) JARA Packages; b) *JARA.cuts* Package; c) *JARA.interf* Package; d) *JARA.heat* Package; e) *JARA.liq* Package; f) *JARA.gas* Package; g*) JARA.phase* Package; h*) JARA.chReac* Package

JARA Architecture

The JARA libraries ("packages" in Modelica terminology) have been organized in order to facilitate their use and maintenance. The package hierarchy is shown in Fig. 1a. JARA is composed of seven packages (see Figs. 1.b - 1.h). The *causal connectors* are defined in the *JARA.cuts* package (see Fig. 1b), and the model interfaces in *JARA.interf* package (see Fig. 1c). The *JARA.gas* package (see Fig. 1f) contains models of control volumes containing gaseous mixtures, of gas transport (i.e., gas-flow by convection and diffusion, valves, pumps, etc.) and boundary conditions (i.e., gas-flow and pressure sources). Similarly, *JARA.liq* package (see Fig. 1e) contains the equivalent models for liquid mixtures. The mixtures of liquids and gases are considered ideal and they can be composed of an arbitrary number of components.

The models related with the heat transport are collected in the *JARA.heat* package (see Fig. 1d): models of control volumes containing solids, thermal resistances and boundary conditions (heat and temperature sources). The *JARA.phase* package (see Fig. 1g) contains models of vapor-liquid phase-change: boiling and condensation. The models of chemical reactions are in *JARA.chReac* package (see Fig. 1h). The modeling details and the library design rules can be found in (Urquia 2000; Urquia and Dormido 2003).

CASE STUDY: BOILER VIRTUAL-LAB

The mathematical model of an industrial boiler is found in (Ramirez 1989). The input of liquid water is placed at the boiler bottom, and the vapor output valve is placed at the top. The water contained in the boiler is continually heated. The diagram of the boiler model, composed using JARA 2i, is shown in Fig. 2a. Two control volumes are considered: (1) a control volume containing the liquid water stored in the boiler; and (2) a gaseous control volume containing the vapor. The vapor volume is equal to the difference between the boiler-recipient inner volume and the water volume.

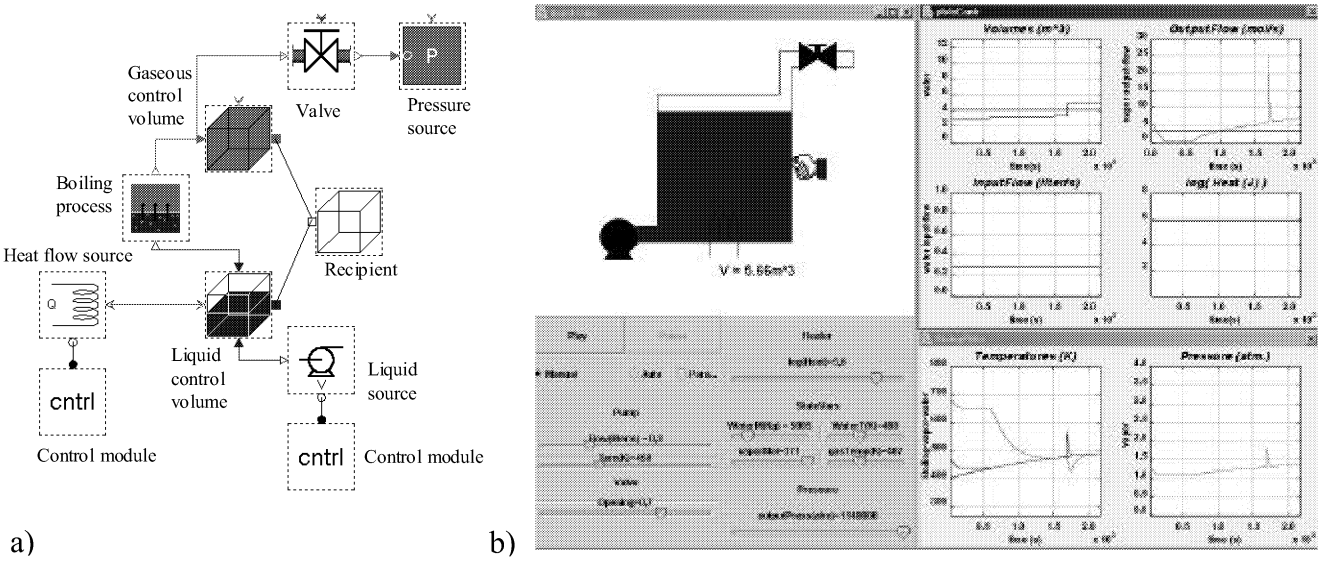


Figure 2: a) Boiler Model Composed Using JARA 2i; b) View of the Boiler Virtual-Lab Implemented using Ejs

The boiling is a transport phenomena represented by a model connecting both control volumes. The heat-flow into de boiler, the pressure at the valve output and the water pump are modeled using JARA source models.

The Ejs view of the boiler virtual-lab is shown in Fig. 2b. The plant has been modeled using JARA 2i. The user can interactively choose between two control strategies: manual and decentralized PID. The control system has been modeled using Simulink: a PID is used to control the water level and another PID is used to control the vapor flow. The manipulated variables are the pump water-flow and the heater heat-flow respectively. The parameters of these PID controllers can be changed interactively. In addition, the value of the model state-variables (mass and temperature of the water and the vapor), parameters (inner volume of boiler), and input variables (temperature of the input water, valve opening and output pressure) can be changed interactively during the simulation run.

CONCLUSIONS

The design and implementation of JARA 2i has been discussed, and its use to develop Ejs virtual-labs has been illustrated by means of a case study. The use of Modelica language has reduced considerably the modeling effort and it has permitted better reuse of the models. Ejs' visual elements have allowed easy creation of the virtual-lab view.

ACKNOWLEDGEMENTS

Part of this work has been supported by the Spanish CICYT, under DPI2001-1012 & DPI2004-01804 grants.

REFERENCES

Bird, R.B; W.E. Stewart and E.N. Lightfoot. 1975. "Transport Phenomena". John Wiley & Sons.

Dynasim. 2002. "Dymola. User's Manual. Version 5.0a". Dynasim AB. Lund, Sweden.

Electric Power Research Institute (EPRI). 1984. "Modular Modeling System. Theory Manual. MMS-02 Release".

Esquembre, F. 2004. "Easy Java Simulations: a Software Tool to Create Scientific Simulations in Java" *Computer Physics Communications*, 156, pp. 199-204.

Himmelblau, D.M. and K.B. Bischoff. 1992. "Process Analysis and Simulation". John Wiley & Sons.

Incropera, F.P. and D.P. DeWitt. 1996. "Fundamentals of Heat and Mass Transfer". John Wiley & Sons.

Martin, C. et al. 2004 "Interactive Simulation of Object-Oriented Hybrid Models, by Combined use of Ejs, Matlab/Simulink and Modelica/Dymola, *In Proc. of the 18th European Simulation Multiconference*, pp. 210-215.

Ramirez, W.F. 1989. "Computational Methods for Process Simulation". Butterworths Publishers, pp. 209-217.

Urquia, A and S. Dormido. 2003. "Object-Oriented Design of Reusable Model Libraries of Hybrid Dynamic Systems". *Mathematical and Computer Modelling of Dynamical Systems*, Vol. 9, No. 1, pp. 65-118.

Urquia, A. 2000. "Modelado Orientado a Objetos y Simulación de Sistemas Híbridos en el Ámbito del Control de Procesos Químicos". Ph.D. Thesis. Dept. Informática y Automática, UNED, Madrid, Spain.

AUTHOR BIOGRAPHIES

CARLA MARTIN received her M.S. degree in Electrical Engineering in 2001 from the University Complutense of Madrid (UCM), Spain. She is currently completing her doctorate studies at UNED, Madrid.

ALFONSO URQUIA received his M.S. degree in Physics in 1992 from UCM and his Ph.D. in 2000 from UNED. His work experience includes six years as an R&D Engineer at AT&T and Lucent Technologies - Bell Labs. Since 2002, he has been working as an Associate Professor at UNED, Madrid.

SEBASTIAN DORMIDO received his Physics degree from UCM (1968) and his Ph.D. from Country Vasc University (1971). In 1981, he was appointed Full Professor of Control Engineering at the UNED Faculty of Sciences. Since 2002, he is President of CEA-IFAC.

SIMULATION IN BIOLOGY

MOTOR OUTPUT VARIABILITY IN A JOINT CONTROL SYSTEM – A SIMULATION STUDY

Oliver Straub
Wolfgang Mader
Jessica Ausborn
Wolfgang Stein
University of Ulm, Dept. of Neurobiology, D-89069 Ulm, Germany
wolfgang.mader@biologie.uni-ulm.de; www.neurobio.de/madSim

KEYWORDS
Physiological simulations, permutation, posture control.

ABSTRACT

The plasticity of the nervous system during the processing of sensory information is of major interest to interdisciplinary neurobiology research. Here, we used an insect model system with stereotyped behavioral patterns for the investigation of how the nervous system can switch between two different motor outputs, walking and standing, despite having the same sensory input, in a computer simulation based on the known network structure. The strengths of 16 specific information pathways which integrate sensory information were permutated and the resulting database of more than 43 million network outputs was analyzed. Two independent analysis show that the same neural network can produce two different behaviors by specifically altering the weighting of several information pathways. We obtained specific combinations of pathway transmission levels that produced these behaviors. This means, that solely changing the strength with which a pathway transmits sensory information is sufficient to switch between different behaviors, like from standing to walking. The predictions that derive from our results can now be used in physiological experiments.

INTRODUCTION

All animal behavior is characterized by its response to sensory stimuli and the movement of the body. The investigation of the processing of sensory information and the control of movements by the nervous system are thus of major interest to neurobiologists, physicists, computer and robotics scientists. Interdisciplinary efforts try to combine the physical structure of the nervous system with the activity of nerve cells and the required computation within the nervous system. Biological systems, especially the vertebrate brain, are very complex and usually possess a large number of nerve cells to produce very flexible behaviors. Therefore, model systems with a lower number of nerve cells and more stereotyped behavioral patterns are used for the investigation of nervous system functions. For walking, the knee of stick insects is used rather than the human knee. While the mechanisms of walking are the same in both species, the number of contributing nerve cells is reduced from several millions in humans to some dozen in insects. In such a small system, it might be possible to answer a fundamental question in neurobiology, namely how a defined set of neurons can produce several different tasks and how to switch between these tasks. Here, the nervous system has to switch from a static joint control during standing to movement control of the limb when the animal starts to walk.
In biological experiments, the process of switching between those two states can often only be investigated by recording the responses of single neurons or a small set of involved neurons at a time. In combination with several degrees of freedom, a meaningful interpretation of experimental data can only be achieved by computational simulations. Here, we investigated the capability of the nervous system of the stick insect to switch between standing and walking with a simulation based on the known network structure. The strength of specific information pathways was permutated and the resulting database of more than 43 million network outputs was analyzed.

BIOLOGICAL BACKGROUND

The neuronal control system of the knee joint (Femur-Tibia joint) of stick insects processes information in three main levels: (1) The sensory information about joint angle (position) and passive or active leg movements (velocity) is measured by the femoral chordotonal organ (Hofmann et al., 1985) and encoded in electrical discharges (spikes). Both sensory parameters are separately transmitted either directly or indirectly via spiking interneurons to (2) a layer of parallel pathways (identified nonspiking interneurons; Büschges 1990, Sauer et al, 1995, Stein & Sauer, 1998), in which the parameter channels are integrated. These nonspiking information pathways finally converge with a context-specific transmission strength onto (3) a single spiking extensor motoneuron (MN), which reflects the previous weighting of position and velocity parameters so that it either contributes to a stiffening of the joint during standing or to active leg movements during walking.
Physiological experiments indicate that for both situations the same underlying set of neurons is computing the sensory information, but leads to opposite motor outputs: a **resistance reflex** that stabilizes the leg during standing or a reflex reversal, the **active reaction** (as defined by Bässler 1976) during walking. Both behaviors can readily be monitored in the spike activity of the motoneuron when a single ramp-and-hold stimulus (i.e. flexion and extension) is applied to the leg. Due to experimental restrictions, it is still unclear whether the known network is sufficient to

produce both motor commands and to switch between them or if additional neurons (of several thousands available in the central nervous system) are needed. We addressed this question in a simulation, which included the structure of the Femur-Tibia network in as much detail as is known from physiological experiments (Fig. 1, an extended version of Sauer et al., 1996). Neurons were modeled with the simulation environment *madSim* (Mader et al., 2003) as a soma with three additional unbranched dendritic compartments with active and passive properties according to Hodgkin-Huxley equations (Hodgkin & Huxley, 1952). The sensory information was separately simulated with current injections to provide position and velocity input to the subsequent information pathways (Büschges, 1994). 16 separate pathways (representing excitatory (E-types) and inhibitory (I-types) nonspiking interneurons) then read out this sensory information so that they comprised different weightings of position and velocity parameters. The differentially computed information conveyed by these pathways was integrated by the extensor motoneuron, which subsequently produced the motor command as a sequence of spikes. We tested whether changes in the transmission strength of these separate pathways can produce both motor outputs and switch between resistance reflex and active reaction.

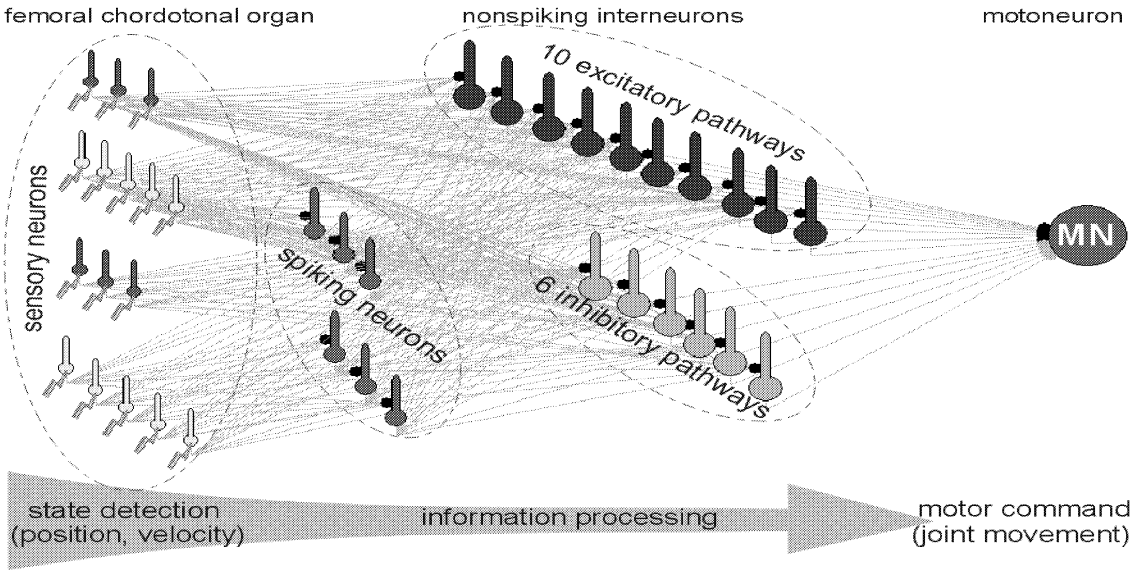


Figure 1: Structure of the Femur-Tibia Control Network and Direction of Information Flow. Small Arrows: Current Injections.

PERMUTATION

Complex biological systems like the nervous system are nonlinear in several levels, like transmitter release, firing frequencies or membrane potential. Therefore, we selected a permutation to test whether the network was capable of generating both, resistance reflex and active reaction. We created a database of motor commands by independently altering the transmission strengths between nonspiking interneuron pathways and the motoneuron. 16 different pathways with three levels of synaptic transmission (low=5 nS, medium=50 nS, high=400 nS) resulted in a database of 43,046,721 motor outputs. To speed up the calculation of the permutations all feedforward parts of the network were calculated only once and the results of this calculation were stored and subsequently used in all following permutations. This reduced calculation time from 175 days to 6 days. Motor outputs tend to stay stable over a broad parameter range, but can also suddenly change with a small alteration of parameters. Methods that reduce the resulting database by driving the system in a certain direction once a sensible motor output is generated, like error back propagation or simulated annealing, would thus increase the probability of losing network results. Additionally, we wanted to study the capability of our network to produce motor outputs without giving it a bias or expectation of what motor output would be meaningful. Instead, we used two tests for scanning the complete database in order to classify motor outputs according to their spike sequence and subsequently look for meaningful results.

Stimulus Dependent Data Analysis

We classified the output of the network by creating a rating system for how close the output resembled (a) resistance reflex or (b) active reaction. (a) Results from physiological experiments were used to determine the optimal spike distribution of a resistance reflex. The simulation time was divided into several sections of different durations according to the ramp-and-hold stimulus. The optimum result was defined with a spontaneous firing activity of 7 Hz, a firing frequency of 90 Hz during the flexion phase, 12 Hz during the holding phase and 0 Hz during the relaxation stimulus. The spike activity of all permutations was compared to the optimum by a rating factor, which was calculated by the sum of the square difference between the current and optimum firing frequency of all time sections.
Similarly, an optimal template was created for the active reaction. A spontaneous firing frequency of 40 Hz was assumed, which was followed by a complete lack of activity beginning 40ms after the onset of the flexion stimulus. When the flexion stimulus reached a certain position, strong firing activity resumed with 90 Hz. All events after the end of the flexion stimulus were not rated.

Stimulus Independent Data Analysis

Our second analysis of the database was stimulus independent: In general, biological neural networks can either produce no, tonic, burst-like (salvo of spikes) or irregular activity. By definition, no, tonic or irregular activity do not represent one of the desired motor patterns. Both, active reaction and resistance reflex are best described by a burst-like activity: In the latter, a single burst is generated with a maximum activity during the flexion part of the stimulus. During an active reaction, two bursts of activity are separated by a lack of spike activity during the flexion part of the stimulus. We searched the database according to the flowchart (Fig. 2) in which the spike activities of the motoneurons were classified in the categories mentioned above.

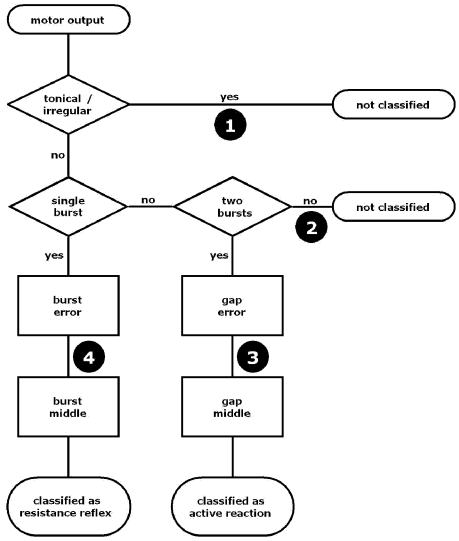


Figure 2: Flowchart for Classification of Motor Outputs

As a result, all database entries without activity or with tonic or irregular firing patterns were eliminated (❶). The remaining bursters were then separated into single bursters or multiple bursters. Multiple bursters were then split into bursters with two bursts or bursters with more bursts, which subsequently were eliminated from our analysis (❷). The remaining single bursters were assigned a variable that held the "burst error", which represents the deviation from an optimum burst duration of 200ms. After that, the center and the duration of the burst were determined and allocated to additional variables (❸).

Simulations with two bursts were assigned the variable "gap error" which held the deviation from a) the optimum duration (100ms) of the lack of activity (which represents the active reaction) and b) from the expected firing frequencies. Subsequently, the center and the duration of the gap were determined (❹).

PERMUTATION RESULTS

Stimulus Dependent Analysis (Resistance Reflex)

All simulation results were rated according to the expected motor output. Fig. 3 shows a comparison of the activity of the best resistance reflex (rating factor 101.80), which fits physiological data very well, and the simulation with the worst rating (rating factor 83,115.00). Bars indicate the synaptic transmission levels of the 16 nonspiking pathways. A plot of the rating factor against the number of simulations sorted by the rating factor revealed that the best 150 simulations showed a steady, but slow rise in rating factor. From simulation 150 on, the rating factor deteriorated more quickly. The regression coefficients of the first 150 and the following 150 simulations were significantly different ($P<0.05$, $N=150$, t-test). We thus grouped the best 150 simulations and classified them as the best possible result. From these simulations, the synaptic transmission strengths of all 16 pathways that relay velocity and position information from the leg onto the motoneuron were used to determine which pathways need to be strong to achieve the desired motor output and which pathways must be weak.

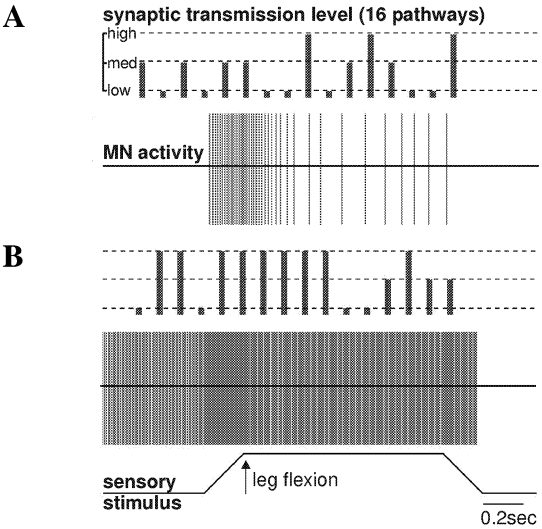


Figure 3: Best (A) and Worst (B) Resistance Reflex

In the best-rated simulations, specific synapses showed a clear preference for a certain synaptic transmission level (Table 1, resistance reflex). If this level was completely unimportant for determining the motor output, one would expect an equal distribution in all pathways, i.e. 33% occurrence for each level (50 in 150 simulations).

Table 1: Occurrence of Synaptic Strengths for different Pathways during Resistance Reflex and Active Reaction

pathway	**resistance reflex**			**active reaction**		
(name)	low	med	high	low	med	high
1 (I4)	55	61	34	**2**	**3**	**84**
2 (E10)	97	53	0	3	3	83
3 (I2)	62	50	38	20	18	51
4 (IE1)	79	31	40	18	23	48
5 (I3)	79	52	19	**29**	**26**	**34**
6 (E6)	61	87	2	32	37	20
7 (E3)	83	65	2	30	51	8
8 (E4)	56	79	15	53	36	0
9 (E5)	69	78	3	54	35	0
10 (E2)	102	48	0	53	36	0
11 (E7)	122	28	0	51	38	0
12 (E8)	131	19	0	41	48	0
13 (E1)	**44**	**58**	**48**	48	41	0
14 (E9)	3	6	141	45	44	0
15 (I8)	2	3	145	48	33	8
16 (I1)	**0**	**0**	**150**	57	32	0

The preference of certain pathways for specific synaptic transmission levels shows the importance of these synapses for the resistance reflex. Two examples, one for a pathway with strong preference (16, nonspiking interneuron I1) and one with a weak preference (13, nonspiking interneuron E1) are shaded.

The difference between the importance of the 16 pathways is most obvious when normalized for each synaptic transmission level (Fig. 4, left). The expected occurrence is 0.33, or, in other words, if all 43 Mio simulation results were plotted, one would obtain an equal distribution of 0.33 for all synaptic transmission levels. As is obvious from our results, the weighting of the different nonspiking pathways is critical for the expression of the resistance reflex.

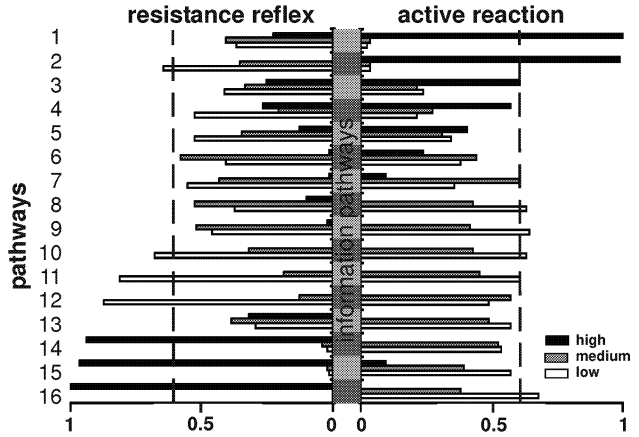


Figure 4: Normalized Occurrence of Synaptic Strengths for all Pathways during Resistance Reflex and Active Reaction

Physiological experiments suggest that the spontaneous activity of the motoneuron may not originate from within the network itself (Bässler, 1983). Our results support this hypothesis, because none of the high rated motor outputs showed spontaneous activity.

Stimulus Dependent Analysis (Active Reaction)

The structure of the simulated network was derived from experimental data in standing animals, and thus during the generation of resistance reflexes. To test, whether the same network is capable of producing the opposite motor output, the active reaction, despite having the same sensory input, we searched the database for active reactions. 89 motor patterns that resembled an active reaction were found. The best result according to the rating factor is shown in Fig. 5.

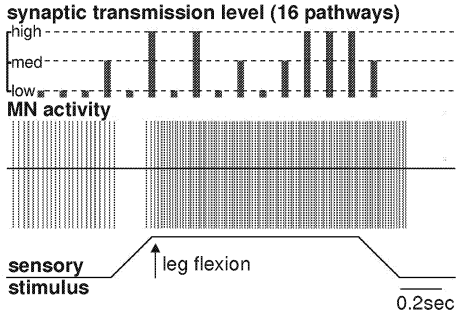


Figure 5: Best Active Reaction with corresponding Synaptic Strengths

Like the experimental original, the simulation showed a high spontaneous activity before the beginning of the stimulus, which was turned off during the flexion phase of the stimulus. Before the end of this stimulus, strong motoneuronal activity resumed.

As with the resistance reflex, the strengths of the 16 pathways differed in their occurrence for this motor pattern (Table 1). Here, a different set of nonspiking pathways became important. Two examples, one for a pathway with strong preference (I4) and one with a weak preference (I3) are shaded (Table 1). Fig. 4 (right) shows which pathways showed preferences for a certain strength and the obvious difference in the occurrences of transmission levels for specific pathways between resistance reflex and active reaction (compare left and right). For example, the highest occurrence of the synaptic transmission level of pathway 2 (nonspiking interneuron E10) changed from low during the resistance reflex to high during the active reaction. This means that this pathway is important for both motor outputs, but needs to be weighted differently. To change the motor output from a resistance reflex to an active reaction, the synaptic transmission level needs to be adapted from the level with the largest bar on the left side to the largest bar on the right side. Going from an active reaction to a resistance reflex requires the reverse procedure.

To test whether the obtained distribution of synaptic transmission levels truthfully drive the network into generating either resistance reflex or active reaction, we calculated 4 permutations with partly fixed synaptic transmission levels. In the first permutation, the synaptic transmission levels of those pathways that had a higher normalized occurrence than 0.6 (dashed line, Fig. 4) during a resistance reflex were fixed to that value, while the transmission levels of all other pathways were permutated (high, medium and low settings). The rating factor for the resistance reflex was used to assess the results. For comparison, a second permutation was calculated in which the synaptic transmission levels of the same pathways were fixed to the value with the lowest probability of occurrence. When we compared the ratings of these two permutations (Fig. 7), the first permutation showed a highly significant ($P<0.0001$, $N=19683$, Mann-Whitney test) better rating than the second permutation, which further supports our hypothesis, that the differential weighting of the pathways is important for the generation of a specific motor command. Keeping synaptic transmission levels of those pathways with high occurrence at the correct value was sufficient and necessary to generate resistance reflexes.

Similarly, we calculated two permutations for the active reaction. The permutation with fixed parameter settings for those pathways with high occurrence produced highly significantly better results ($P<0.0001$, $N=2187$, Mann-Whitney Test) than the second permutation. This shows that also for the active reaction the weighting of the nonspiking pathways is of great importance and that a specific weighting is sufficient and necessary to produce an appropriate motor output.

Stimulus Independent Analysis

The stimulus independent analysis (Fig. 2) resulted in more than 350,000 single burst (resistance reflex like) and more than 650,000 two burst (active reaction like) motor patterns.

The best rated resistance reflex and the best rated active reaction of the *stimulus dependent* analysis were present in the best 0.1% of the results.
For comparison with the stimulus dependent analysis, the 150 best rated single bursters and the 89 best rated two bursters were analyzed in detail: The single bursters had an average burst duration of 307.67±13.73 ms (n=150) and burst-centers between 655 ms and 688 ms simulation time. This corresponded very well with the stimulus dependent analysis. The occurrence of synaptic transmission levels of these resistance reflex like motor pattern showed a clear preference of specific synapses for a certain strength (Fig. 6, left).

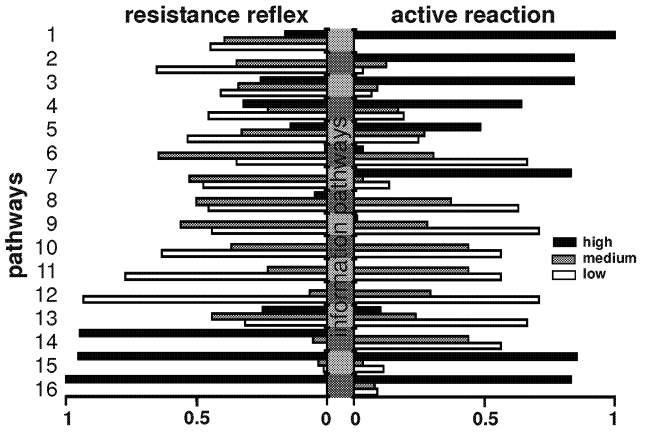


Figure 6: Normalized Occurrence of Synaptic Strengths according to the Stimulus Independent Analysis.

As is obvious, the weighting of the different nonspiking pathways was very similar to the result of the *stimulus dependent* analysis (compare to Fig. 4, left). Thus, even an analysis that did not involve the stimulus protocol detected resistance reflexes and showed that the weighting of the different pathways is critical for the expression of this motor output.
The best "two bursters" had an average burst-interval of 72.43±9.53 ms and the center of the interval was always between 513 ms and 643 ms. Although the interval appeared earlier than in the stimulus dependent analysis and the average interval was shorter, the motor outputs clearly resembled an active reaction. The occurrences of synaptic transmission levels (Fig. 6, right) showed great similarities with those obtained by the stimulus dependent analysis (Fig. 4, right). The most noticeable differences were pathways 15, 16 and 7, which changed from low to high synaptic strengths and from medium to high, resp. We conclude that these pathways do not strongly contribute to the reversal of the motor output, but may rather determine the timing of the active reaction and caused it to appear earlier and to be of shorter duration.

CONCLUSIONS

Our analysis of the permutation results shows that the same neural network which stabilizes the leg during standing can also contribute to active leg movements, despite having the same sensory input. To do so, the weighting of several information pathways which integrate sensory information about joint velocity and position must be altered specifically. We have shown which interneuronal pathways need to be changed for switching between both behaviors and that these changes are sufficient and necessary for producing these behaviors. How the change in the transmission of sensory information to the motoneuron is actually achieved was not tested, but it is conceivable that such a change would either occur directly presynaptic or directly postsynaptic (like in our simulation) to the nonspiking interneurons. Our results predict that specific interneuronal pathways need to be weakened or strengthened (Table 1) to support standing or walking. The timing of the motor output seems to depend on the weighting of the nonspiking pathways. These predictions now can be tested in physiological experiments.
In our investigation, we did not reduce the number of possible simulations by using a goal-oriented permutation. Instead, we tested whether the generation of a certain behavior is an intrinsic property of this network, which turned out to be the case. Although we did not start up with an expectation for synaptic transmission levels, we obtained specific combinations of synaptic transmission levels that produced specific behaviors. This means that solely by changing the strength with which a pathway transmits sensory information is sufficient to switch between different behaviors, like from standing to walking.

REFERENCES

Bässler, U. 1976. "Reversal of a reflex to a single motoneuron in the stick insect Carausius morosus." *Biological Cybernetics,* No. 24, 47-49.

Bässler, U. 1983. *Neural Basis of Elementary Behavior in Stick Insects.* Springer Verlag, Berlin:

Büschges, A. 1990. "Nonspiking pathways in a joint-control loop of the stick insect Carausius morosus." *Journal of Experimental Biology,* No. 151, 133-160.

Büschges, A. 1994. "The physiology of sensory cells in the ventral scoloparium of the stick insect femoral chordotonal organ." *Journal of Experimental Biology,* No. 189, 285-292.

Hodgkin, A.L. and A.F. Huxley. 1952. "A quantitative description of membrane current und its application to conduction and excitation in nerve." *Journal of Physiology,* No. 117, 500-544.

Hofmann, T.; U.T. Koch and U. Bässler. 1985. "Physiology of the femoral chordotonal organ in the stick insect, Cuniculina impigra." *Journal of Experimental Biology,* No.114, 207-223.

Mader, W.; J. Ausborn; O. Straub and W. Stein. 2003. "madSim - a tool for simulating biological neuronal networks." *Proceedings of the 29th Göttingen Neurobiology Conference 2003* (Göttingen, Germany, Jun.12-15). Thieme Verlag, Stuttgart, 1055-1056.

Sauer, A.E.; R.B. Driesang; A. Büschges and U. Bässler. 1995. "Information processing in the femur-tibia control loop of stick insects. 1. The response characteristics of two nonspiking interneurons result from parallel excitatory and inhibitory inputs." *Journal of Comparative Physiology. A,* No. 177, 145-158.

Sauer, A.E.; R.B. Driesang; A. Büschges and U. Bässler. 1996. "Distributed processing on the basis of parallel and antagonistic pathways simulation of the femur-tibia control system in the stick insect." *Journal of Computational Neuroscience,* No. 3, 179-198.

Stein, W. and A.E. Sauer. 1998. "Modulation of sensorimotor pathways associated with gain changes in a posture-control network of an insect." *Journal of Comparative Physiology. A,* No. 183, 489-501.

COMBINING SIMULATED BIOLOGICAL COMPONENTS TO ACHIEVE A POWERFUL SIMULATION ENVIRONMENT

Michael Schmid and Jens Kohlmeyer
Department of Programming Methodology and Compiler Construction
University of Ulm
89069 Ulm, Germany
michael.schmid@informatik.uni-ulm.de
jens.kohlmeyer@informatik.uni-ulm.de

ABSTRACT

In this paper the approach to achieve a powerful simulation environment by combining simulated singular biological components is shown. The singular components like neurons, biological sensors or articulations are modelled and simulated each on its own with hybrid tools used in computer science. On the basis of these parts a widespread simulation environment can be composed to form a powerful tool which can be used in science and education to examine for example feedback loops with complex simulation runs.
The different components have been developed separately, so we consider multifaceted ways of combining them.

INTRODUCTION

In the past we made efforts to model and simulate complex biological structures and their behavior with software engineering tools. In spite of some minor difficulties it became obvious that hybrid tools such as the combination of Simulink und Stateflow from The Mathworks, Inc. are very well suited to model biological components and simulate their behavior.
So far we modelled two biological components: the axon of a nerve cell (see [Kohlmeyer et al., 2003]) and the motion control of an insect leg (further information in [Raschke, 2003]). In the first part of this paper the third component, a biological sensor, is presented. This paper consists of two parts. In the first part we describe our latest modelled biological component, a sensor called muscle spindle. The second part deals with the combination of the components.

MODELLING A MUSCLE SPINDLE

The biological sensor is a neuromuscular spindle of a vertebrate skeleton muscle. The neuromuscular spindle was selected, because there is no simulation based on ionic channels available for this sensor and the output of this sensor can be used as an input for an axon ([Kohlmeyer et al., 2003]).
The model is composed of four hierarchical layers as shown in figure 1. The four levels are introduced from top to bottom in the following.

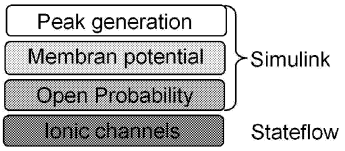


Figure 1: The layers of the model

Peak generation

The peak generation layer is responsible for calculating and displaying the number of action potentials (APs) per second. APs are generated from the potential of the cell membrane. The data for this generation are the results from several physiological measurements (see for example [Kakuda, 2000]). In figure 2 a model of the muscle spindle is shown. On the bottom there are two different types of fibres pictured: *Ia* and *II* fibres. The *Ia fibres* are called primary (fast) and they detect velocity. The *II fibres* are called secondary (slow) and they detect length and position.

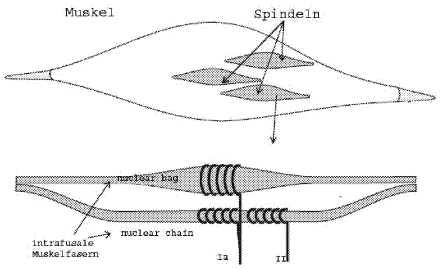


Figure 2: Model of a neuromuscular spindle

Membrane potential calculation

The Goldmann equation (see formula (1), [Kohlmeyer et al., 2003]) is used to calculate the membrane potential from the ion currents, which are generated by the ionic channels. The sensitive endings of the neuromuscular spindle afferents (*Ia/II fibres*) consist of mechanical gated ionic channels. The opening of these ionic channels changes the permeability of the cell membrane towards several selective ions. These ions now may easily pass the membrane and generate a current. For more details see [Alberts, 2002]. For simplification in this model only the Na^+ ion current and the K^+ ion current were taken to calculate the change of the membrane potential with the Goldmann equation.

$$V_m = \frac{RT}{F} * ln \frac{P_K[K^+]_a + P_{Na}[Na^+]_a + P_{Cl}[CL^-]_i}{P_K[K^+]_i + P_{Na}[Na^+]_i + P_{Cl}[Cl^-]_a} \qquad (1)$$

Calculating the probability of gating

In the biological reality the ionic channels of a muscle spindle show a flickering behavior [Alberts, 2002]. This means the channels are controlled by change. In layer three this natural behavior is modelled.
Each channel has it's own characteristics. To represent this characteristics, the adaption time for each channel is set to a random number and each channel has a specific range on which it normally opens. All the random numbers which are involved in the opening decision of a channel are calculated in each simulation step. First a fixed number for each channel is taken from a pre generated list. This number represents the preferred point of gating for one channel. A random generated number is added to the fixed number.

Next step is the calculation of the open probability of the ionic channels. A change of the angle in the crook of the arm was used in this simulation. Changes in this angle are converted to a force (f). This force is applied against the mechanosensitive ionic channels. Each channel has two states: open and close. The energy difference between the two states, Δg, dictates the likelihood of the channel occupying each state, as described by the Bolzmann distribution (2) [Prof.William Brewer,].

$$\frac{P_o}{P_c} = exp(-\Delta g / k_B T) \qquad (2)$$

After a few calculations, this results in the final equation (3). For further information please see [Sukharev and Corey, 2004]. The open probability p_o is now a function of the force f which results of changes in the angle mentioned above.

$$P_o = \frac{1}{1 + exp(-fb + \Delta u)/(k_B T)} \qquad (3)$$

The final probability of gating(p_o) is compared to the random number which represents the preferred point of gating. In case the random number is lower or equal than p_o, the channel opens.

Ionic channels

The general basis of the model are the two types of ion channels. At the sensitive endings of the *Ia* and the *II fibres* only mechanical gated ion channels are involved. During a stretch of a muscle the neuromuscular spindles are also stretched, and with them the intrafusal fibres. This puts stress to the membrane surface of the *Ia* and *II fibres* in which the ion channels are located. In response to this stimulus the channels open and ion currents are generated. The two types of fibres have a different behavior. The primary (Ia) fibres are fast adaptive while the secondary adept slowly [Hille, 1992]. Based on this knowledge two types of ionic channels were implemented. On the basis of the fast channels the bottom layer (the Stateflow level) is now exemplified.

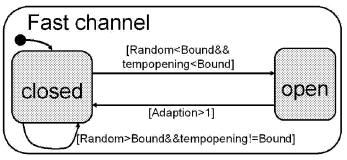


Figure 3: A primary ionic channel

Figure 3 shows a Stateflow model of a fast primary ionic channel. It consists of two states, *open* and *closed*. The behavior of the fast channels towards a stretch stimulus of the membrane is implemented as described below.
Corresponding to a random number and a calculated opening probability (bound) the channel opens. This generates an amount of ion current which is passed to the Goldmann equation. The channel is open for a specified adaption time and then closes. Each channel has its own adaption time. The initial value for the calculation of the adaption time was taken from a random number generator. A change in the force (calculated from the angle) means a change in the open probability. The channel makes his decision to open only if the open probability increases. To achieve this, the open probability of the last simulations step is stored in a variable. Apart from this opening of a ionic channel, in each state there is a small probability of spontaneous opening (in state *closed*) or closing (in state *open*). In this way the random behavior is considered (see [Kohlmeyer et al., 2003]).

The slow secondary ionic channels have been modelled analogically, of course adapted to their specific behavior.

Putting all described parts together results in a simulation model of a muscle spindle. As stimulus a force which is calculated from a change of the angle is used. This stimulus puts stress to the membrane surface, which results in gating of the mechanosensitive ionic channels. The open channels are generating an ion current which is passed to the Goldmann equation. The Goldmann equation calculates a membrane voltage, from which an AP frequency is calculated.
Figure 4 shows a typical ramp hold situation simulated with the model. This ramp hold situation is an example of an experiment with muscle spindles (for detailed description please see [Denocker et al., 2003]).
The membrane potential of the primary (1) and secondary (2) endings on a ramp hold stretch of four degree angle movement (3) are displayed. The primary fast channels are in advance of the secondary. The primary are adapting shortly after the hold phase has started. The secondary channels are rising constantly till the start of the hold phase and without any adaption during the hold phase.

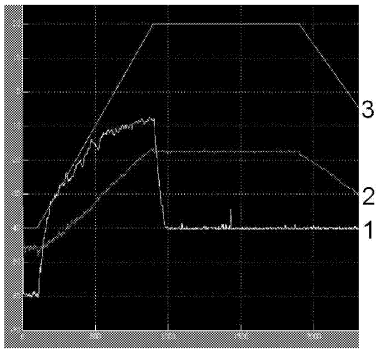


Figure 4: Ramp and hold simulation (screenshot)

COMBINATION AND EXTENSION

A general interface definition for the usage in further extensions was made. The definition is based on biological systems and their cooperation in nature. Each module should provide and handle the data like their natural paragons. In this case the axon gets a electric potential as an input and the spindle generates an electrical potential. In addition to this definition a vector is added to the interface for additional non natural occurrence data. The interfaces of both models were implemented to fit this definition. The model of the axon (see [Kohlmeyer et al., 2003]) was attached to the output potential of the spindle. The two modules worked surprisingly very well together. For the *II fibres* the calculated action potentials and the simulated ones are nearly fitting, except of some shift because the simulation consumes more time than the calculation. Because of machine limitations the number of ionic channels in each model was reduced. For the *Ia fibres* still some kind of synchronization has to put between the models to work together. To fix the synchronization and the limitation problem we are currently evaluating a distributed system approach where each model is simulated on a separate machine and the communication is done via network. Currently we are able to connect two running simulations (the axon and the spindle) on different machines. The resulting simulation environment works very well and fast.

CONCLUSION

In this paper we introduced a new component of our simulation environment, the model of a biological sensor called muscle spindle. We described a possible way of combining these singular components to achieve a powerful simulation environment which can be used in education and science. The understanding of complex biological processes is highly benefited by using such a tool.

References

[Alberts, 2002] Alberts, B. (2002). *Molecular biology of the cell.* Garland Science.

[Denocker et al., 2003] Denocker, L., Picquet, F., Petit, J., and Falempin, M. (2003). Characterization of Spindle Afferents in Rat Soleus Muscle using Ramp-and-Hold and Sinusoidal Streches.

[Hille, 1992] Hille, B. (1992). *Ionic channels of excitable membrans.* Sinaver Associates.

[Kakuda, 2000] Kakuda, N. (2000). Response of human muscle spindle afferents to sinusoidal stretching with a wide range of amplitudes.

[Kohlmeyer et al., 2003] Kohlmeyer, J., Sarstedt, S., and Mader, W. (2003). Modelling and simulation of a biological process (neural action potential) with hybrid tools used in computer science. Proceedings of ESM2003, p. 328-332.

[Prof.William Brewer,] Prof.William Brewer, P. Das elektrische Potential.
http://www.physik.fu-berlin.de/~brewer/vm_elpot.html.

[Raschke, 2003] Raschke, A. (2003). Einsatz von Werkzeugen der Informatik zur Modellierung und Simulation von Bewwegungssteuerung und Katalepsie im Insektenbein. Masterthesis Universität Ulm.

[Sukharev and Corey, 2004] Sukharev, S. and Corey, D. P. (2004). Mechanosensitive channels: Multiplicity of families and gating paradigms. Science STKE.

Neural Network Based Model to Flow Cytometric Data Interpretation

Pereira, G. C.
COPPE/UFRJ - Federal University of Rio de Janeiro
Bonomo, A.
Experimental Medicine Division, Research Coordination, Cancer National Institute
Ebecken, N.F.F.
COPPE/UFRJ - Federal University of Rio de Janeiro

KEYWORDS
Flow Cytometry; Neural Network; Diagnostic.

ABSTRACT

Flow cytometry is a largely employed method in the research and medical practice where their usages as diagnostic exam on grave diseases like infections by HIV viruses and mainly leukemia. This paper aims new methodological approach. Two artificial neural network based on supervised training are applied to pattern recognition of their optical blood cell population coming from patients. The preliminary results demonstrate this approach is promising and can contribute soon to protocols related to this disease.

INTRODUCTION

Flow cytometry are equipments which use light dispersion and quantify different cellular populations (Van Dan, 1999).. Depending of the arrangement it can obtain two or thirteen simultaneous parameters for each cell or organism. Many diseases depend of right "classification" from flow cytometers to accurated diagnostic and subsequent treatment. Patients infected, for instance by HIV viruses needs their CD4+ cells quantified to suitable evolution and treatment. The determination of a DNA index (which reveals the chromosome number of a tumoral cell in relation to normal) in malign cells is important as a prognostic factor of tumoral evolution. The "imunophenotype" and classification of cellular subtypes are crucial to establish the suitable treatment (Pui, 1997). However, the cytometric data analysis based on bi-dimensional visualization, limits the reachness of this technique, see figure I.

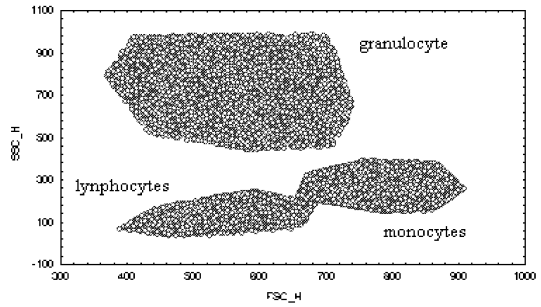


Fig. 1 – Three type blood cells: granulocytes, monocytes and lymphocytes

In addition the subjectivity of the method depends on expertise and "good sense" of operator who set, how and which cell populations are important. At this context, the application of neural networks became considerable because it is able to take the whole information in side the multi-dimensional space of data segregating it accordingly previously establish patterns.

CELULAR KIND DETERMINATION

There is no doubt about the cytometry contribution to biological studies mainly on cell identification in tissue of organism.
The periphery blood is built of three principal constituents: lymphocyte, monocyte, and granulocyte. The central idea at this first approach is whether each population has self-characteristics, which differentiate themselves from other. This characteristics act on the optical properties generated by cytometers giving own "signatures" of populations.

Type 1					
	FSC-H	**SSC-H**	**FL1-H**	**FL2-H**	**FL3-H**
max.	739	987	538	479	503
min.	369	441	0	0	0
mean	558.9	724.2	211.6	215.4	222.7
s.d.	78.32	125.72	39.80	40.39	36.66
Type 2					
	FSC-H	**SSC-H**	**FL1-H**	**FL2-H**	**FL3-H**
max.	684	906	362	392	402
min.	390	33	0	0	0
mean	546.1	119.7	142.9	129.3	110.3
s.d.	54.44	37.85	52.59	52.72	54.23
Type 3					
	FSC-H	**SSC-H**	**FL1-H**	**FL2-H**	**FL3-H**
max.	908	387	405	427	591
min.	659	146	0	0	0
mean	782.8	262.3	213.9	195.9	180.6
s.d.	52.72	52.20	41.88	46.72	51.93

Table I – Basic statistics of cytometer readings

Nevertheless, depending on physiological conditions under which these cells are submitted or their different degrees of

cellular development, these signatures also vary contributing to great variability of signals yielded by the equipment, as shown in table I.
Even the same phenotype variations of the same population give some oscillations on medical patterns and can became evident subpopulations. Thus the object of this work is to study the capability of artificial neural networks to "learn" these variabilities to affect the data base with five optical parameters generated from three cited cell types.
Initially, was an acquired sample of periphery blood at the Cancer National Institute (INCA). The three blood cell populations were isolated. It was used a Fac Scan Becton DiKson flow cytometer equipped with a single blue laser of 488 nm. Such samples were passed through the laser beam cell by cell, and the light scatter parameters determined by their forward scatter, side scatter and three fluorescences (F1, F2, and F3), see figure II and retreated in list mode.

Each numeric value is the shining intensity received by the sensors. The resulting matrix is built of five parameters (columns) and each row related to each cell (event). We have performed 10000 readings for each kind of cell separately. Later, an additional column was settled showing the type 1 to lymphocyte, 2 to monocyte and 3 to granulocyte and their optical variabilities given under "exiting" conditions. Following, the three dataset was joint together performing a single matrix with six columns (parameters) and thirty thousands row (cases).

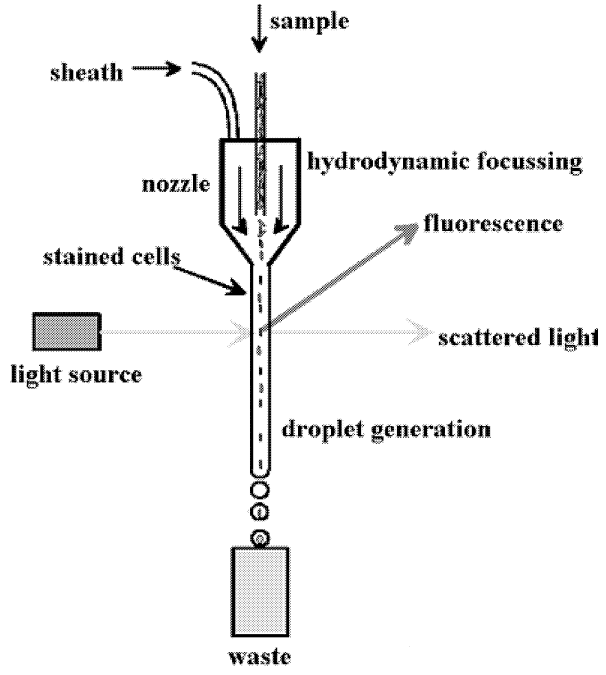


Figure II – Flow Cytometry principle

NEURAL NETWORK TYPES

Artificial Neural networks (as compared to real ones) are mathematical systems that are comprised of a number of "processing units" that are linked via weighted interconnections. A processing unit takes weighted signals from other neurons, sums them up, transforms them and outputs a numeric result. Processing units are often considered crudely analogous to real neurons and since they are linked together in a mesh or network, the name Neural Networks was coined.
Many neural networks have their neurons structured in "layers" that have similar characteristics and execute their transfer functions in synchronization (at the same time, virtually speaking). Nearly all neural networks have neurons that accept data and neurons that produce outputs.
The behavior of neural networks, how they map input data to output data, is influenced primarily by the transfer functions of neurons, how they are interconnected and the weights of those interconnections.
Typically, an architecture or structure of a neural network is established and one of a variety of mathematical algorithms is used to determine what the weights of the interconnections should be to maximize the accuracy of the outputs produced. Neural networks are "trained", meaning they are previous example to establish (learn) the relationships between the input variables and the predicted variables by setting these weights. Once these relationships are established (the neural network is trained), the neural network can be presented with new input variables and it will generate predictions.

BACK PROPAGATION NEURAL NETWORK (BP)

Back Propagation neural networks are probably the most common neural architecture implemented today (Rumelhart, 1986). Back Propagation neural networks are "Multilayer Feed Forward Networks" meaning they have a layered architecture where data flows into one layer, through one or more "hidden" (internal) layers and out an output layer. Data flows in one direction, thus "Feed Forward" and has multiple layers, thus "MultiLayer". Back Propagation is actually an iterative gradient decent learning algorithm used to set the weights. The error between the neural network outputs and desired outputs are propagated backwards through the network to aid in the determination of what the new values of weights should be. This is done for each training data record over and over until the error reaches some stable minimum.

RADIAL BASIS FUNCTIONS

The radial Basis Functions (RBF) architecture generates a mapping between inputs and outputs using a weighted sum of basis functions (Bishop, 1991 and 1995). The basis functions are radial Gaussians (i.e., the basis functions are invariant to rotations about their centers). Free parameters include the Gaussian center locations, the spread (standard

deviation) of each Gaussian, and the weights which determine the degree to which Gaussian contributes to each output.

An RBF model can be trained by an iterative gradient descent method similar to the Back propagation algorithm used for training the MLP. However, empirical results have shown that RBF can be trained much faster using a two-step process, without much degradation in performance. The two steps are:

1. Determine the centers and spreads of the Gaussians by minimizing the average variation of call training patterns to Gaussian centers using *K* Means clustering;

2. Adjust the weights from each Gaussian to the outputs by minimizing the mean square error. Since the weighted-sum relationship between an output variable and the Gaussian is linear, the global optimum can be computed quickly (via the use of pseudo-matrix inversion to solve a standard least-squares problem).

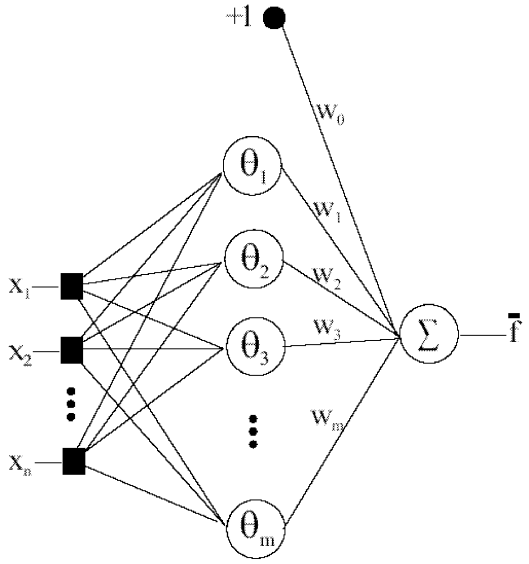


Figure III - RBF network architecture

NETWORK ARCHITECTURE OPTIMIZATION

Once it was possible to have data from isolated populations, the problem now falls over what is the most fitted neural network architecture to solve this classification. At the supervised training we have tested two systems: Multilayer Perceptron Neural Network (MLP) and Radial Basis Function Neural Network (RBF). In both cases, the so called back-propagation algorithm was used (Holland, 1975 and Opitz, 1997). The optimization of number of computational layers, number of neurons in each layer, connectivity degree between neurons and what would be the best set of input variables, a genetic algorithm, a heuristic search technique, was left to evolve until the maximum of two hidden layers to avoid high computational costs. All computational experiments usually employed 50% of data for training, 25% for testing and 25% for validation. The composition of this data profile was randomly chosen. In another step, other data base having a mix of signatures of the three populations but at this time without the indicating column was submitted to the best models.

The training of a RBF network can be seen as an optimization problem. If the centers and widths of the hidden RBF`s are not fixed, it turns into a difficult nonlinear task. In the case where the number of units in the hidden layer is to be optimized the problem becomes still harder, with a non-differentiable cost function. In such cases, using a genetic algorithm (Holland, 1975 and Goldberg,1989) as an optimization tool is a natural choice. There are several possibilities of using a GA to configure a RBF network. A straightforward approach is to fix the topology and use the GA as an optimization method to compute all free parameters. It has beem done by Sheta for time series forecasting.

In Mak (1998), the number of hidden units was fixed, and the GA optimized only the location of the centers. The widths of the functios and the outputs weights were computed by the K-nearest neighbor heuristic and SDV (singular value decomposition), respectively. Whitehead and Choate (1995) also fixed the number os RBF`s, and evolved their locations and widths, but on a completely different and interesting fashon: instead of encoding an entire model in each individual, the whole set of chomosomes cooperate to constitute a RBF network, each one being a basis function. Whitehead (1996), presents an alternative approach to evaluate each individual, based on SVD.

However, using a GA to configure only the parameters of the RBF network does not seem to be the best choice, specially considering there are specific (and faster) algorithms to do this task. So, a better alternative would be to adopt the GA to also determine the network topology. The literature offers a large number of works with this proposals. In this way it seems to support this hypothesis. Whitehead and Choate (1993), Billings (1995), Carse (1995), Maillard (1997), Liu (1999), Zao (2000), de Lacerda (2000), and Barreto & Ebecken (2002)are some good examples. In these approaches, the GA is responsible for setting the number of hidden units, as well as the location of their centers and their widths, while the weights in the output layer are computed by an algebraic or gradient-descent method. Naturally, there are many differences between them. In Whitehead and Choate (1993), for example, an indirect representation is used; the locations of centers are governed by space-filling curves, whose parameters evolve genetically. Another example is the work of Maillard and Gueriot (1997) in which the basis function are not restricted to Gaussians and are also subjected to evolution. Based in statements given by Zao (2000), the work of Barreto & Ebecken (2002) employs a minimalist initialization procedure within the GA, which

makes the evolution process somewhat similar to connectionist constructive methods.
Another idea is a hybridize the configuration process, using the GA as a support tool for another algorithm. In Burdsall (1997), the GA was used to optimize the number and initial cluster centers of the K-means algorithm; the RBF network training then proceeded as in Moody (1989).
Chen et al. (1995, 1999) have present a two-level hybrid learning method for RBF networks which employs both GA`s and the OLS algorithm. A regularized orthogonal least squares algorithm (ROLS) was employed to construt the RBF networks at the inner level, while the two main parameters of this algorithm, the global RBF`s width and the regularization parameter were optimized by a GA process at the outer level.

RESULTS

Many computational experiments were done and the six best models (three MLP and three RBF) separated and its results compared following table II.
In general, all models shows good robustness and discriminative capabilities between the three classes.
It was allowed the model evolution until a maximum of twenty computational units in hidden layer. It is easy to verify that MLPs had good correlation between the true class and those indicated by the models (0,95 %). The model MLP-3 shows the worst performance just it have to use 18 hidden neurons to get almost the same result of MLP-1 which have only one hidden neuron. It is worth to see, both models were nor able to optimize their respective architecture using all five input variables. However, MLP-2 with only three inputs and one hidden unit reach better performance with RMSE 0,175380 and AME of 0,246011 contrasting this class of models. Actually, MLP-2 was able to classify all 30000 events saving four unknown cases.

-NN type	Layers	Correlation	RMSE	AME
MLP-1	05:01:01	0.955364	0.301196	0.246011
MLP-2	03:01:01	0.955147	0.244372	0.175380
MLP-3	05:18:01	0.953830	0.369072	0.314557
RBF-1	02:12:01	0.985542	0.138341	0.097539
RBF-2	02:04:01	0.993029	0.151542	0.131009
RBF-3	02:05:01	0.985542	0.138341	0.097536

Table II: Model obtained results

When analyzing Radial Basis Function Neural Networks an interesting case could be noted about their performance. It is clear this type of model have better performance (correlation of 0,99%, 0,98%) while MLP (0,95%). Really, RBF-1 shows twenty hidden neurons, correlation of 0,98%, RMSE 0,138341and AME of 0,097539, but RBF-3 smaller architecture (2:5:1), the same numerical correlation and RMSE yet minor AME (0,097536), possibly due to its topological optimization. This fact gives an evidence of several neural network can reach similar results in spite of their architectures, the difference in this case stays on the higher or lesser computational effort.
Whatever the same table, any one can see the model RBF-2 which get the best correlation index amongst whole models reaching the brand of 0,99%. Its RMSE is 0,151542 and AME 0,131009 displaying an excellent performance even so numerically “inferior” to those RBF-1 and RBF-3. This architecture was able to place all studied events at their concerning classes and present yet the best approximation.

DISCUSSION

As much the systems MLP as RBF are able to classify flow cytometry patterns with plenty accuracy. The problem denote very simple despite a large optical variability of each cell population be linearly separated and this data base can be segmented though networks only with the forward and side scatter parameters. The promising potential application of neural network in order to detection of Minimal Residual Disease (MRD) in leukemias by the persistence of a small number of sick cells after the first treatment cycle have primordial prognosis importance and will determine the evolution and subsequent treatment. Detection of MRD through molecular biology technique, PCR in real time, is unequivocal (Panzer-Grümayer et al., 2000), but its cost is extremely high, forbidden nowadays. The custon option is flow cytometry, which use phenotype characteristic of leukemia cells (Counstan-Smith, 1998; Campana, 2000; (Counstan-Smith, 2000). With the utilization of fluorescent markers of many wave length, besides physical cell characteristics given by light dispersion parameters, will be possible the identification of specific molecules of sick cells and characterize their phenotype. However, leukemic cells come from normal cells of osseous medulla and in most cases have very similar phenotypes. In such case, using the conventional cytometric analysis, from bidimensional visualization an overlap of two populations exists, causing false MRD. At this context, the multivariate analysis of neural network capabilities can be a good aid.

REFERENCES

A. Barreto, H. Barbosa, N. Ebecken, Growing compact RBF networks using a genetic algorithm, in: Proc. of the VII Brazilian Symposium on Neural Networks, Recife, PE, 2002, pp. 61–66.

B. Burdsall, C. Giraud-Carrier, GA-RBF: A self-optimising RBF network, in: Proc. of the Third International Conference on Artificial Neural Networks and Genetic Algorithms, Springer-Verlag, 1997, pp. 348–351.
URL citeseer.nj.nec.com/71534.html

Billings, S.A., G.L. Zheng, Radial basis function network configuration using genetic algorithms, Neural Networks 8 (6) (1995) 877-890.

C. Bishop, 1991, “Improving the Generalization Properties of Radial Basis Function Neural Network”, Neural Computation, 3(4): 579-588.

C. M. Bishop, 1995, “Neural Networks for Pattern Recognition”, Clarendon Press, Oxford.

Carse, B. A.G.Pipe, T.C.Fogarty, T.Hill, Evolving radial basis function neural networks using a genetic algorithm, in Proc. Of the IEEE Int. Conf. On Evolutionary Computation, IEEE, 1995, pp. 300-305, volume 1.

Coustan-Smith E, Behm, F.G., Sanchez, J., Boyett, J.M., Raimondi, S.C., Rubnitz, J.E., Riviera, G.K., Sandlund, J.T., Pui, C.H., 1998. “Campana, D. Imunological Detection of Minimal Residual Disease in Children with Acute Lymphoblastic Leukemia”, Lancet: 351(9102): 550-4.

Coustan-Smith E., Sanchez, J., Hancock, M.L., Boyett, J.M., Behm, F.G., Ribeiro, R.C., Pui, C.H., Campana, D. 1998. “Clinical Importance of Minimal Residual Disease in Childhood Acute Lymphoblastic Leukemia”. Blood; 96(8): 2691-2696.

Coustan-Smith E., Sanchez, J., Hancock, M.L., Boyett, J.M., Behm, F.G., Ribeiro, R.C., Pui, C.H., 2000. “Campana, D. Clinical Importance of Minimal Residual Disease in Childhood Acute Lymphoblastic Leukemia”. Blood; 96(8): 2691-2695.

E. G. M. de Lacerda, A. C. P. L. F. de Carvalho, T. B. Ludermir, Evolutionary optimization of RBF networks, in: VIth Brazilian Symposium on Neural Networks, Sociedade Brasileira de Computac¸ ˜ao (SBC), IEEE, 2000, pp. 219–224, volume I.

E. P. Maillard, D. Gueriot, RBF neural network, basis functions and genetic algorithm, in: Int. Conf. on Neural Networks, IEEE, 1997, pp. 2187–2192, volume 4.

G. P. Liu, V. Kadirkamanathan, Multiobjective criteria for neural network structure selection and identification of nonlinear systems using genetic algorithms, in: IEEE Proc. Part P: Control Theory and Applications, 146(5), 1999, pp. 373–382.

Goldberg, D. Genetic algorithms in search, optimization, and machine learning, Addison-Wesley Readings, MA, 1989.

Holland, J.H., 1975, “Adaptation in natural and Artificial Systems, Ann Arbor, The University of Michigan Press.

K. O. Stanley, R. Miikkulainen, Evolving neural networks through augmenting topologies.
URL citeseer.nj.nec.com/stanley01evolving.html

Mak, M.W., K.W. Cho, Genetic evolution of radial basis function centers for pattern classification, in: Proceedings of The 1998 IEEE International Joint Conference on Neural Network, 1998, pp. 669-673, volume 1.

Moody, J., C. Darken, Fast learning in networks of locally-tuned processing units, Neural Computation 1 (2) (1989) 281-294.

Optiz D. Shavilik, J, 1997, “Connectionist Theory Refinement: Genetically Searching the Space of Network Topologies, Journal of Artificial Intelligence Research, 6, pp. 177-209.

Panzer-Grümayer E.R., Schneider, M., Panzer, S., Fasching, K., Gadner, H., 2000. ”Rapid Molecular Response During Early Induction Chemotherapy Predicts A Good Outcome In Childhood Acute Lymphoblastic Leukemia”. Blood; 95(3): 790-794,

Pui, C.H. 1997. “Acute Lymphoblastic Leukemia”. Pediatric Clin North Am.; 44(4):831-846.

Rumelhart, D.E., Hilton, G.E. and Willians, R.J., 1986, “Learning Internal Representation by Error Propagation”, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations (edited by D.E. Rumelhant and J.L. McClelant). MIT Press, Cambridge, MA.

Q. F. Zhao, O. Hammami, K. Kuroda, K. Saito, Cooperative co-evolutionary algorithm how to evaluate a module, in: IEEE Symposium on Combinations of Evolutionary Computation and Neural Networks, 2000, pp. 150–157.

S. Chen, Y.Wun, K. Alkadhimi, A two-layer learning method for radial basis function networks using combined genetic and regularised OLS algorithms, in: Proceedings of Genetic Algorithms in Engineering Systems: Innovations and Applications, Vol. 414, IEEE, 1995, pp. 245–249.

] S. Chen, Y. Wu, B. L. Luk, Combined genetic algorithm optimization and regularized orthogonal least squares learning for radial basis function networks, IEEE-NN 10 (5) (1999) 1239.

Sheta, A.F., K.D. Jong, Time-series forecasting using GA-tuned radial basis functions, Imformation Sciences Special issue.

Van Dam P.A., Tjalma W.A.A., 1999, “Clinical Applications of Flow Cytrometry”. In Recent Advances in Histopatology, 18, Lowe D6, Underwoord JCE (eds).

Whitehead, B.A., T.D. Choat, Cooperative-competitive genetic evolution of radial basis function centers and widths for time series prediction, IEEE Trans. On Neural Networks 7(4)(1995) 869-880.

Whitehead, B.A., Genetc evolution of radial basis function coverage using orthogonal niches, IEEE Trans. On Neural Networks 7 (6) (1996) 1525-1528.

Whitehead, B.A., T.D. Choat, Evolving space-filling curves to distribute radial basis functions over an imput space, IEEE Trans. On Neural Networks 5 (1) (1993) 15-23.

ANALOG MODULATION OF DIGITAL COMPUTATION IN NERVE CELLS: SIMULATING THE STOMATOGASTRIC NERVOUS SYSTEM OF THE CRAB.

Wolfgang Stein and Jessica Ausborn
Abteilung Neurobiologie
Universität Ulm
D-89069 Ulm, Germany
E-mail: wstein@neurobiologie.de

ABSTRACT

In modeling studies, the performance of neural networks is often attributed to the complexity of the structure of the network and the number of connections between neurons. In contrast, recent research in neurobiology shows that an analog modulation of the connections between neurons and of the integrative properties of single neurons can strongly enhance the network performance and might even be capable of replacing parts of the network without losing the task-specific output. We modeled the stomatogastric nervous system of the crab, using the biologically oriented network simulator *madSim* to test the impact of such analog modulations on network function. This network has been studied intensively and provides a solid data base for a simulation.
Our model shows that the analog modulation of a single neuronal connection determines the output of the whole network. Also, an analog computation within a single neuron can substitute missing network parts and rescue the desired function.

INTRODUCTION

The performance of neural networks usually is thought to depend mainly on the number of neurons and their connections. This is because in most modeling studies, neurons are seen as digital devices that compute the digital information from other neurons (Sima and Orponen 2003). Plasticity is usually handled as a variation in the number or the strength of the connections (including long term potentiation and depression). However, the nervous system is much more than just a collection of connections. Recent research in neurobiology has shown that analog modulation of the digital processing is at least as important as the network structure. Neurons use digital signals mainly to conduct information across large distances. Active and passive properties of the receiving neurons and their synapses are, however, very flexible and enable even a rigidly connected network to show high plasticity. Especially the influence of modulatory substances in the bloodstream or from modulatory neurons plays a major role in determining the output of the network. Experiments show that with the same connectivity, it is possible to generate completely different motor outputs, while on the other hand the same motor output can be generated with completely different connectivities (Prinz et al. 2003).
Most simulations of biological neuronal networks focus on the structure of the network. Only for single cells or small numbers of cells the influence of intrinsic neuronal properties and an analog modulation of the connections between neurons has been studied. The possibility to insert a large amount of computation into single cells opens a new dimension for the modeling of neural networks that needs to be investigated in more detail.
Here, we used the neuronal circuits in the crustacean stomatogastric nervous system to investigate whether an analog computation within single cells can substitute the computation of whole network structures and to test the impact of an analog modulation of the digital connection between neurons on the generation of the motor output of a rather complex neural network.

BIOLOGICAL BACKGROUND

General considerations

Neurons are generally subdivided into three parts (Fig. 1): (1) A receiving element, the dendrite, (2) a main compartment, the soma and (3) a long process called the axon. The soma is usually physically distant from the next neuron and is connected to it via the axon and the output synapse. Information on the axon is coded in the frequency of all-or-nothing electrical discharges (action potentials). In the axon terminal, just before reaching its target, these action potentials shift to a more analog mode, because they are no longer actively propagated. Rather, they deteriorate in amplitude. The amplitude that is left when the action potential finally reaches the connector to the next neuron (the synapse) determines the amount of chemical transmitter that is released onto the follower neuron. The transmitter then elicits graded (analog) postsynaptic signals that passively (with decrement) travel along the receiving dendrite towards the soma, where they are summed up and elicit new action potentials.

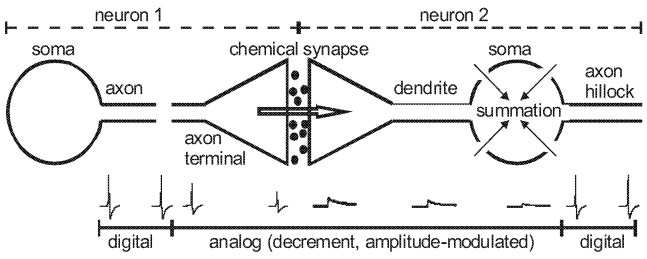


Figure 1: Information Transmission and Processing within Neurons.

The summation of inputs strongly depends on the intrinsic properties of the soma, which are nonlinear. These properties can change the occurrence (frequency) and the timing (time of appearance) of the action potentials that are then relayed to the next neuron.

The stomatogastric nervous system

The stomatogastric nervous system of the crab is a well-known model system for central pattern generation (e.g. Nusbaum and Beenhakker 2002) and has been the subject of several modeling studies (e.g. Nadim et al. 1998). Overlapping sets of neurons comprise the gastric mill and pyloric circuits, which generate the slow gastric mill rhythm and the rapid pyloric rhythm (Fig. 2A, summary: Nusbaum and Beenhakker 2002). The gastric mill rhythm controls the chewing of food with internal teeth in the stomach of the crab, while the pyloric rhythmic determines the filtering of food.
The half-center circuit of the chewing rhythm consists of a reciprocal inhibitory connection between the lateral gastric mill motoneuron (LG) and Interneuron 1 (Int1, Fig. 2B). It oscillates upon release of excitatory input from projection neurons like the modulatory commissural neuron 1 (MCN1). LG inhibits the axon terminal of MCN1 (T1). This presynaptic inhibition modulates the digital information from MCN1 before it reaches LG. The contribution of the presynaptic inhibition to pattern generation is yet unclear. It is believed that the network connections mainly determine the generation of the chewing rhythm (Bartos and Nusbaum 1997).
In addition to the modulation of inputs, intrinsic neuronal properties affect the generation of the chewing and the filtering rhythms. With these properties active, chewing rhythms that are similar to the MCN1-elicited rhythm can be produced even in the absence of MCN1 and of most network connections (Hertzberg et al. 2003; Brown and Wood 2003). This suggests that also intrinsic properties may contribute to the generation of the motor output. It is, however, unclear whether they are functional in intact animals.
An experimental investigation of a) the influence of the modulation of action potential amplitude by presynaptic inhibition and b) the impact of intrinsic neuronal properties, like voltage dependent ion channels, on the motor output is very difficult because the half-center circuit cannot be completely isolated from the rest of the network. In the model, we were able to take a closer look at the contribution of presynaptic inhibition and intrinsic neuronal properties.

SIMULATION

We modeled the neuronal network underlying the generation of two different motor commands, the filtering and the chewing rhythms (Fig. 2A), to test (1) the contribution of presynaptic inhibition to rhythm generation as an example for an analog modulation of neural connections (* in Fig. 2A) and (2) the capability of analog nonlinear computations in consequence of intrinsic cellular properties (** in Fig. 2A) to produce rhythmic motor output.
The network was built according to electrophysiological findings. Hodgkin-Huxley-like neurons (Hodgkin and Huxley 1952) were used for constructing the projection neuron MCN1 and the core pattern generator neurons of the chewing rhythm, LG and Int1. MCN1 was equipped with separate synaptic terminals for electrical (T2) and chemical (T1) transmission. Terminals were passively conducting elements with appropriate properties for analog information processing (Fig. 2B). A simple filtering network was constructed, which produced a stable triphasic rhythm that was similar in period, phase relationships and duty cycles (the relative duration of neuron activity during a period) to the rhythms obtained from electrophysiological recordings (Fig. 2C).

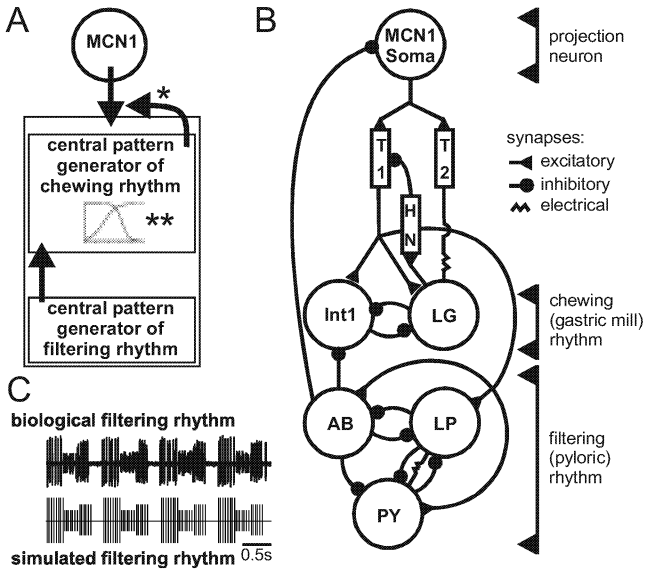


Figure 2: Overview of the Connectivity in the Stomatogastric Nervous System.

The different synaptic inputs to LG differ in strength and time course and were thus adjusted according to experimental data. MCN1's firing frequency was set to 20 Hz, in agreement with experimental data. We modeled the physiological effects of the inhibition of the MCN1 axon terminal by connecting the MCN1 axon to a nonspiking element with graded transmitter release. This 'terminal' then received inhibition from LG which decreased action potential amplitude. The inhibition could be varied in strength and time course. To create the appropriate time delay and duration of the presynaptic inhibition, we added a helper neuron (HN). This helper neuron did not possess any voltage-dependent currents and acted as a passive compartment of the LG neuron, or an axon terminal of LG.

TESTING THE ACCURACY OF THE MODEL

In the animal, the period of the chewing rhythm is regulated by the filtering rhythm by means of the inhibitory connection between AB and Int1 (Fig. 3A). We used this regulation of the chewing rhythm to test our model.

In Fig. 3 we compared the simulation results of the gastro-pyloric interaction to experimental data: Like in the animal, the chewing rhythm was affected by the filtering rhythm (Fig. 3B). With the filtering rhythm active, the period of the chewing rhythm decreased, which was mainly due to a shortening of the LG interburst interval (as in the animal). LG burst duration remained unchanged.

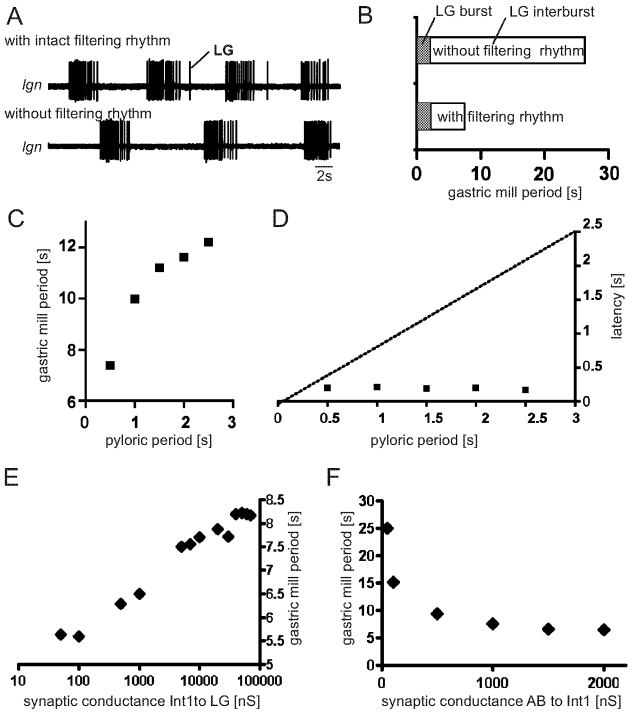


Figure 3: Gastro-Pyloric Interactions.

The period of the chewing rhythm depended on the period of the filtering rhythm (Fig. 3C). Each LG burst onset was time-locked to the preceding burst of the pyloric neuron AB at all pyloric periods (Fig. 3D), which shows that the LG burst onset depended on the LG disinhibition from Int1. Consequently, a strengthening of the synapse from Int1 onto LG (Fig. 3E) slowed down the chewing rhythm and a stronger synapse between the pyloric neuron AB and Int1 supported disinhibition of LG and thus sped up the rhythm (Fig. 3F).

In conclusion, gastro-pyloric circuit interactions of the modeled network proved to be comparable to the properties of the biological rhythms and truthfully reproduced experimental data (Bartos et al. 1999).

ANALOG MODULATION OF NEURAL CONNECTIONS

The presynaptic inhibition onto the terminal of MCN1 caused a slow, but long-lasting decrease of MCN1 spike amplitude after the LG burst (Fig. 4, *). The middle trace of Fig. 4 shows the passively conducted action potentials (lower part clipped) in the MCN1 terminal. LG oscillated and produced bursts of action potentials, which caused an increase in presynaptic inhibition in the MCN1 terminal (*). LG oscillations were stable over a broad range of synaptic conductances of the presynaptic inhibition. The oscillations disappeared and LG started to fire tonically when presynaptic inhibition was removed from the system.

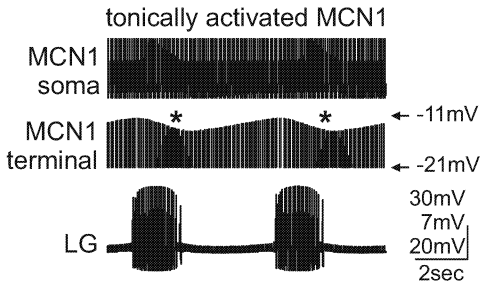


Figure 4: Effects of LG Spike Activity and Presynaptic Inhibition on the MCN1 Terminal in the Model.

With intact presynaptic inhibition, the period and the phase relationships of the two half-center neurons depended on the *strength* and *duration* of the inhibition (Fig. 5A,B). The *duration* of the presynaptic inhibition was adjusted by changing the capacity of the helper neuron. An increase in capacity caused a slower build-up and reduction in strength of the presynaptic inhibition, which in turn prolonged the period of the chewing rhythm (Fig. 5A). This was due to an increase in the duration of the activity (burst duration) of both, LG and Int1. Also, Int1's duty cycle was prolonged with the increase in period, indicating that Int1 burst duration increased faster than LG burst duration (not shown). The *strength* of the presynaptic inhibition was changed by either altering the synaptic conductance of the synapse from the helper neuron to the MCN1 terminal or the synaptic conductance from LG to the helper neuron. Both manipulations produced similar results. With stronger presynaptic inhibition, the period of the chewing rhythm increased, although in this case, LG burst duration got shorter (Fig. 5B).

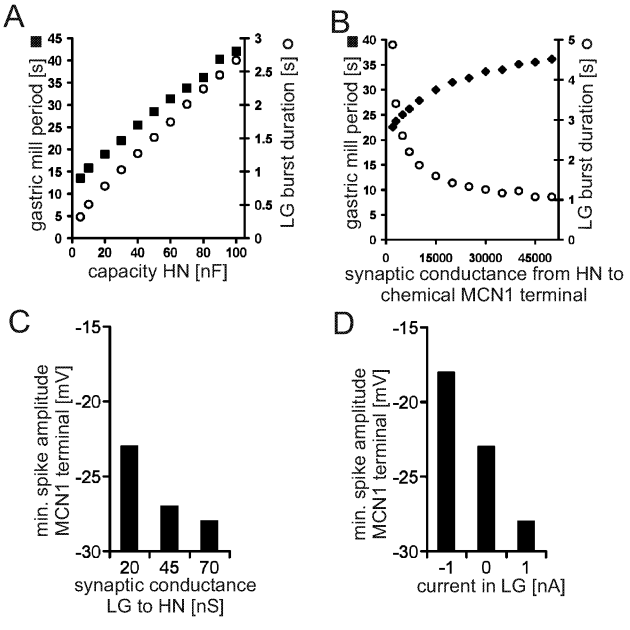


Figure 5: Two Ways to Alter Presynaptic Inhibition.

In electrophysiological experiments, the possibilities to change the presynaptic inhibition are limited. One possible experiment is to de- or hyperpolarize LG and change its firing frequency, which is expected to have an impact on presynaptic inhibition. The impact of such a manipulation on presynaptic inhibition and the two rhythms is, however, unclear. We used our model to test whether both, a change in the strength of the presynaptic inhibition induced by a change of synaptic conductance and current injections into LG had similar effects. When looking at the decrease of spike amplitude in the MCN1 terminal, both manipulations had indeed similar effects (Fig. 5C,D).
With respect to the rhythm, the effects differed (Fig. 6A,B): When the synaptic conductance was increased, the rhythm slowed down, due to a strong increase in Int1 burst duration and duty cycle, while LG burst duration and duty cycle decreased (Fig. 6A). In contrast, when LG was depolarized, the rhythm sped up and LG burst duration and duty cycle increased. Consequently, the burst duration of Int1 and its duty cycle decreased (Fig. 6B).

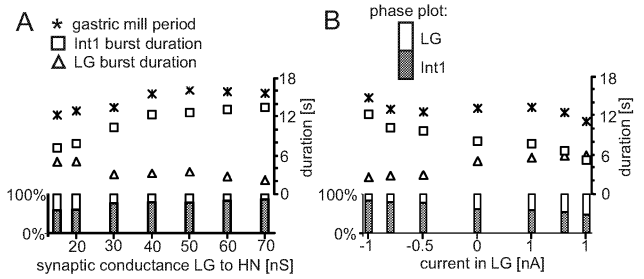


Figure 6: Period and Timing of the Chewing Rhythm Depended on the Way, how Presynaptic Inhibition was Altered.

These results show that the presynaptic inhibition is indeed a critical factor for the generation of the chewing rhythm. Presynaptic inhibition determines the burst durations of the neurons and the period of the rhythm despite an unchanged network structure.
Our model also shows that the speed of the filtering rhythm depended on the strength of the presynaptic inhibition (Fig 7): Interestingly, both, a direct enhancement of the presynaptic inhibition by means of the synaptic conductance (Fig. 7A) and current injections into LG (Fig. 7B) had similar effects on the filtering rhythm (in contrast to the chewing rhythm, see above): Both manipulations caused a slowing of the filtering rhythm during the LG activity (burst) as well as during the LG interburst.

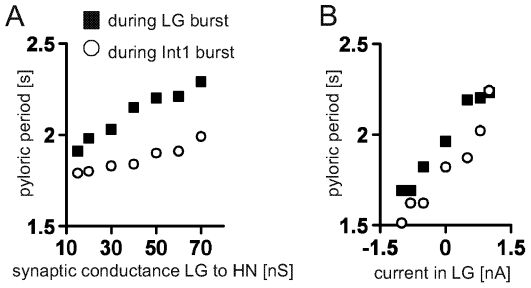


Figure 7: Both Ways of Altering Presynaptic Inhibition had Similar Effects on the Filtering Rhythm.

These results show that the chewing rhythm and the filtering rhythm can be generated with the correct network connections, but the network outputs strongly depended on the analog decrease of a single synaptic transmission by means of presynaptic inhibition. The model emphasizes the importance of an analog modulation of the digital connections between neurons.

ANALOG COMPUTATION WITHIN SINGLE CELLS

In the stomatogastric nervous system, many neurons possess the inherent capability to produce long-lasting bursts of activity (plateau potentials) or a rebound after being inhibited (postinhibitory rebound). These effects are due to specific regenerative currents through ion channels in the cell membrane of these neurons and enable them to produce whole bursts of action potentials even after an inhibitory input. These ionic currents represent an enhanced analog computation of synaptic inputs. In the central pattern generator of the chewing rhythm, LG activity might be supported by a plateau-like potential (Brown and Wood 2003). It is, however, unclear, whether these potentials are functional in behaving animals. We tested whether or not the implementation of such regenerative currents could substitute parts of the computation that is contributed by the interconnections of neurons in the network.
In the model, the effects of additional regenerative membrane properties on LG activity can easily be monitored. To this end, the model was reduced to only the half-center neurons and the filtering network. We provided LG with several additional user defined-currents (according to Buchholtz et al. 1992) to enable postinhibitory rebound and plateau potentials. For postinhibitory rebound, a hyperpolarization-activated current (I_h, Fig. 8A,B) and a transient inactivating calcium current (I(Ca1), Fig. 8C) was added. Steady-state activation (a_∞) and time constant (τ) of I_h are given in Fig. 8A, those of I(Ca1) in Fig. 8C. An increasing hyperpolarizing current injection into LG elicited a stronger sag of the membrane potential and a stronger rebound (Fig. 8B).

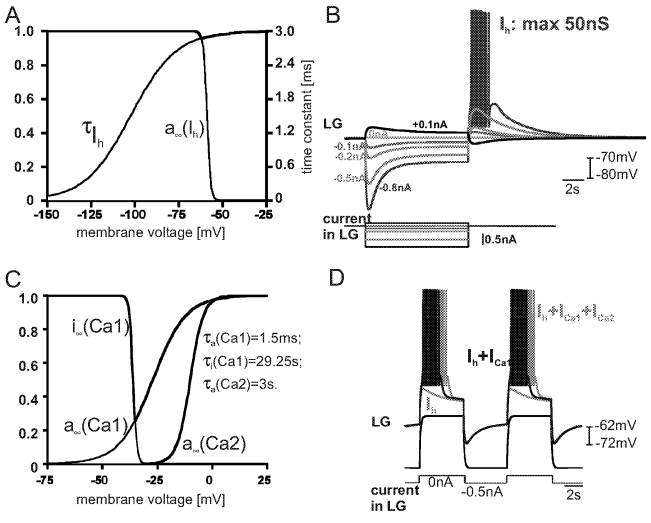


Figure 8: Nonlinear Ionic Currents and Resulting Output of Analog Computation in LG

A noninactivating calcium current (I(Ca2)) enabled plateau potentials (for steady state activation (a_∞) and time constant see Fig. 8C, for current details see Buchholtz et al. 1992.). With all currents, LG showed a postinhibitory rebound with subsequent plateau potential and spike activity after a hyperpolarizing current injection (Fig. 8D) or inhibitory synaptic events.

To establish oscillations of the half-center, Int1 was tonically depolarized with current injections to raise it above spike threshold. However, half-center oscillations only appeared when Int1 received rhythmic synaptic inhibition from the pyloric neuron AB, which in turn caused a disinhibition of LG with subsequent rebound and plateau potential (Fig. 9). Without this input from the filtering network, no chewing rhythm was present. This finding was confirmed by electrophysiological experiments (Hertzberg et al. 2003).

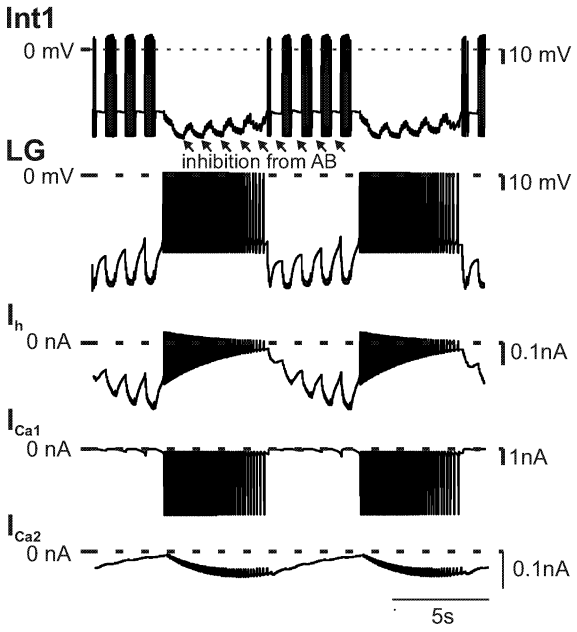


Figure 9: Half-Center Oscillations of Int1 and LG without MCN1.

Our model shows that even with most network connections impaired, a solid rhythm can be generated. This highlights the importance of intrinsic analog computations within single neurons for the whole network output.

CONCLUSIONS

We have shown that our model truthfully reproduced the properties of the chewing and the filtering rhythms and the interaction between them.

The model predicts that only in the presence of presynaptic inhibition from LG to MCN1, the network properties were sufficient to elicit a gastric mill rhythm. Both, an increase in synaptic conductance and an increase of LG activity led to a stronger presynaptic inhibition and had similar effects on the filtering rhythm. Both modification, however, led to opposite changes in the chewing rhythm.

Without MCN1, half-center oscillations could not be elicited unless LG was provided with a postinhibitory rebound and plateau potentials and Int1 received rhythmic inhibition from the filtering rhythm.

With respect to the comparison of the analog and digital computing in neural networks, the model shows that the analog modulation of the digital connection between neurons by means of presynaptic inhibition determines the network output. Two simple parameters (duration and strength) of the modulation are the determining factors.

Also, when most of the network connections are removed, an analog computation within a single neuron can substitute the computation of the missing network parts and rescue the motor pattern. This reflects the high degree of redundancy in structure and function which can be observed throughout the nervous system.

For the future, we aim to investigate redundancies in the nervous system in physiological experiments based on the results of the present study.

REFERENCES

Bartos, M. and M.P. Nusbaum. 1997. "Intercircuit control of motor pattern modulation by presynaptic inhibition." *Journal of Neuroscience*, No.17(7), 2247-2256.

Bartos, M.; Y. Manor; F. Nadim; E. Marder and M.P. Nusbaum. 1999. "Coordination of fast and slow rhythmic neuronal circuits." *Journal of Neuroscience*, No.19(15), 6650-6660.

Brown, B.E. and D.E. Wood. 2003. "Induction and regulation of rhythmic bursting by a peptidergic projection neuron." *Proceedings of the Society for Neuroscience 33* (New Orleans, LA, Nov.8-12). Washington, DC, 500.7.

Buchholtz, F.; J. Golowasch; I.R. Epstein and E. Marder. 1992. "Mathematical model of an identified stomatogastric ganglion neuron." *Journal of Neurophysiology*, No.67(2), 332-340.

Hertzberg, S.R.; J. Huybrechts; L. Schoofs and M.P. Nusbaum. 2003. "Distinct mechanisms underlie similar patterns from the same circuit." *Proceedings of the Society for Neuroscience 33* (New Orleans, LA, Nov.8-12). Washington, DC, 604.14.

Hodgkin, A.L. and A.F. Huxley. 1952. "A quantitative description of membrane current and its application to conduction and excitation in nerve." *Journal of Physiology*, No.117(4), 500-544.

Nadim, F.; Y. Manor; M.P. Nusbaum and E. Marder. 1998. "Frequency regulation of a slow rhythm by a fast periodic input." *Journal of Neuroscience*, No.18(13), 5053-5067.

Nusbaum, M.P. and M.P. Beenhakker. 2002. "A small-systems approach to motor pattern generation." *Nature*, No.417, 343-350.

Prinz, A.A.; V. Thirumalai and E. Marder. 2003. "The functional consequences of changes in the strength and duration of synaptic inputs to oscillatory neurons." *Journal of Neuroscience*, No.23(3), 943-954.

Sima, J. and P. Orponen. 2003. "General-purpose computation with neural networks: a survey of complexity theoretic results." *Neural Computation*, No.15(12), 2727-2778.

AUTHOR BIOGRAPHY

WOLFGANG STEIN is currently working with the stomatogastric nervous system of crustaceans at the University of Ulm, Germany. www.neurobiologie.de.

JESSICA AUSBORN is working on her PhD thesis at the University of Ulm, Germany. Her thesis is concerned with the simulation of the locust flight pattern generator.

CARDIAC RESYNCHRONIZATION THERAPY BY BIVENTRICULAR STIMULATION: ANALYSIS AND MODELS OF PATIENT MANAGEMENT

Sara Bertolino, Stefano Pasinato and Giorgio Romanin-Jacur
University of Padova
Stradella San Nicola 3
36100 Vicenza VI, Italy
E-mail: romjac@dei.unipd.it

KEYWORDS
Heart Failure, Pharmacological Therapy, Resynchronization Therapy, Patient Management, Simulation Model.

ABSTRACT

Heart Failure with Left Bundle Branch Block is a disease consisting in asynchrony between right and left ventricle contraction, which causes strong reduction in the amount of pumped blood; it is considered a disabling and progressive disease which dramatically compromises life quality and expectation. Such disease may be treated by pharmacological treatment or by Heart Resinchronization Therapy, obtained by implantation of a special pacemacker equipped with three electrocatheters, which effect Biventricular Stimulation, thus obtaining determinant improvements of heart functions, and consequent disease reduction. We have built a simulation model of patient management, based on actual data obtained from Vicenza hospital, in order to evaluate patient history in terms of ambulatory care, hospital admissions and possible synchronization therapy, in order to improve patient routes for what concerns both clinical results and involved costs.

HEART FAILURE AND LEFT BUNDLE BRANCH BLOCK

Heart Failure (HF) is defined as heart inability to supply the amount of blood required by human body to satisfy metabolic exigencies which are necessary to perform typical everyday life activities. It may occur either in acute or in chronic form; chronic heart failure, which will be considered here, is associated with a continuous disease progression and is accompanied by more and more frequent flare-ups.
In the most common shape heart failure causes left ventricle enlargement and contraction power reduction, with consequent left ventricle pump deficiency: as time goes by this condition originates a series of processes which increase heart damage and related symptoms.
About 30% of patients suffering from advanced heart failure present Left Bundle Branch Block, which further worsens clinical conditions, as it causes inter-intra ventricular delay: because of a lack of electrical conduction, left ventricle contraction and relaxation delay with respect to the same right ventricle functions, and the consequent asynchrony reduces heart pump effect.
Heart Failure with Left Bundle Branch Block is considered a disabling and progressive disease which dramatically compromises life quality and expectation.
In Europe at least ten million people suffer from heart failure, and that amount appears to be increasing. Heart failure represents the principal cause of hospitalization and one of major causes of disability for patients over 65.
Without going into details, heart failure may be caused by different pathologies; it proceeds through different phases: an initial myocardial damage, which indices a ventricle malfunction, then a series of adaptive and compensating responses, finally the clinical condition of full-blown heart failure.

HEART FAILURE DIAGNOSIS AND THERAPY

Heart failure may be diagnosed by means of:
- patient history analysis, based on the presence of the most common symptoms, like dyspnea, debility, etc,
- physical examination, which includes vital signes analysis (arterial pressure, cardial frequency, breathing, body temperature), cardiovascular system analysis (jugular vein, heart size, heart sound), breathing system analysis, neurological inspection, etc.,
- diagnostic and laboratory tests, like general inspection, thorax X-ray, electrocardiogram, echocardiography, ematochemical tests, electrophysiological tests, stress test.

In particular an electrocardiogram can detect left bundle branch block by measuring QRS length, i.e., the duration of ventricle contraction, which is between 0.05 and 0.10 seconds for a healthy body, and larger for a sick one, with different trace. Echocardiography can determine ventricle ejection fraction, i.e., the ratio between the amount of blood which is expelled during contraction and the amount of blood inside the ventricle: for a healthy body it is greater than 50%.
Heart failure severeness is classified by the New York Heart Association according to four levels, characterized by symptoms presence: NYHA I, symptoms present only under stress (22% of people suffering from heart failure), NYHA II, during normal activity (43%), NYHA III, during limited activity (33%), NYHA IV, during rest (2%).
Heart failure therapy includes:
- health education, aimed at involving the patient by diet, social habits, relax and physical exercise,
- pharmacological treatment, which includes intake of different medicines, at low or high dosage, either orally or by intravenous injection, according to disease advance state and to general patient conditions,
- surgical operations, which include myocardic revascularization, pacemaker or similar instrument implantation, and heart transplantation.

Heart Resinchronization Therapy is obtained by a special pacemacker equipped with three electrocatheters, which effect Biventricular Stimulation, i.e., they give temporized electrical impulses to heart atrium and to left and righ

ventricle, so that the two ventricles contract simultaneously, at a predetermined delay after atria contraction; such a way interventricular delay is dramatically reduced and the amount of pumped blood increased. Generally such a therapy is associated with pharmacological treatment. Patients suitable for heart synchronization therapy are selected according to the following criteria:

- they are classified as NYHA III or NYHA IV,
- QRS length is over 0.13 seconds,
- ventricle ejection fraction is under 35%,
- they are refractory to pharmacological treatment.

Heart synchronization therapy is applied in Europe since 1998. Studies on the matter are reported by Braunsweigh et al., 2000 and by Curnis et al., 2003.

PATIENT MANAGEMENT MODEL

We built up a model to represent patient management, i.e., all possible routes followed by patients through hospital departments to get diagnises and associated treatments, in order to evaluate times and costs involved, and to suggest possible changes able to improve results in terms of better clinical effects and cost reduction.

We considered 1515 patients suffering from heart failure, of which 236 selected for resynchronization therapy, admitted at San Bortolo hospital in Vicenza (Italy), where such a therapy is effected, between January 2001 and September 2002. From patient documentation a random sample of 100 patients was extracted to obtain good estimates of admission durations, inter-admission times and probabilities of movements among different hospital departments and different disease states.

A patient may be admitted to hospital after a medical examination by the general practitioner or by the specialist (planned or unplanned admission), or alternatively through the emergency service after an attack; he/she is generally admitted in one of the departments of Fig. 1, according to general state and presented symptoms; admissions to other departments are rare and generally due to attendant pathologies; in every considered department he/she is diagnosed and consequently treated or moved to a different department; patients admitted in Cardiology may be moved to the Electrophysiological Application Unit to get resynchronization therapy when suitable.

To build the model four attributes of the patient were taken into account:

- patient state: S1 = NYHA I and II (35%), S2 = NYHA III and IV not selected for resynchronization (44%), S3 = NYHA III and IV selected for resynchronization (21%);
- therapy: TF1 = low dosage pharmacological treatment, TF2 = high dosage pharmacological treatment, CRT = cardial resynchronization therapy;
- diagnostic procedure: D1 = general inspection + electrocardiogram + ematochemical tests to identify state S1; D2 = D1 + echocardiography + X-ray, to identify S2; D3 = D2 + coronarography, to identify S3;
- admission ward.

The model in its MicroSaint implementation is reported in Fig.2, where tasks (simulating activities and parameter settings) are represented by nodes, routes imposing task sequences are represented by arrows and finally choices of routes out of a task are represented by diamonds at the right of task nodes. Task Start gives the starting input, task Paziente generates patients, tasks S1, S2 and S3 give attributes to related patients, tasks in the third column simulate admission intervals in the different wards and tasks in the fourth column simulate therapies and compute related costs; task Out simulate dismission and follow up and task Ritorno the time at home between two sequential admissions. The choice of the route to be taken, after a task activity is performed, is ruled by probability P, which depends on patient state and on effected diagnosis (possibly mistaken and leading to unsuitable therapy). Routing probabilities and residence times (gamma distributions) have been obtained from inspected actual patient clinical reports.

Results from simulation runs include the total length of admissions during two years of treatments and two following years of follow up, and the involved costs. For any type of route followed by a patient, total cost is the sum of single costs due to diagnosis, therapy, admissions, follow up and emergency; costs have been analytically computed.

In the current situation the mean number of admission days is 52.40, of which 23.77 during the first two years of treatment; the difference is mainly due to the fact that the majority of patients is cured by pharmacological treatments, which reduce symptoms but do not avoid aggravation and crises; note that in particular the mean number of admission days is 55.66 for those patients which are apt to resynchronization but are cured by pharmacological treatments. The mean cost is 13,003 Euro.

A second series of simulation runs considers the modified situation where patients suffering from heart failures are all admitted to the cardiology ward, without any exception, and only the ones in state S1 may then be moved to other wards: such a way the probability of mistaken diagnosis and consequent unsuitable therapy strongly decreases, according to the specialization of the ward and to the more advanced instrumentation. In such a case all admitted patients virtually selected for resynchronization are effectively implanted and dramatically improve their clinical condition. The resulting mean cost reduces to 11,554 Euro; the cost reduction is due to savings consequent to a high decrease of the amount of admission days and of pharmacological treatments for patients in state S3, which overcome the total implantation cost. In particular if we focus the analysis to patients in state S3, their mean cost is 17,819 Euro in the current situation and reduces to 14,640 Euro in the modified one; we must consider that we have considered only direct costs and we have neglected indirect ones associated with low physical capacity and consequent necessity for external help; moreover non quantifiable costs are associated with low life quality. Obviously management changes inducing the above modified situation were strongly recommended.

REFERENCES

Braunschweig F., Linde C., Gadler F., Ryden L., 2000, "Reduction of hospital days by biventricular pacing", *European Journal of Heart Failure,* february.

Curnis A., Caprai F., Bontempi L., Scivales A., Bianchetti L., Nodari S., Dei Cas L., 2003, "Valutazione economica della resincronizzazione cardiaca nei pazienti affetti da scompenso cardiaco moderato-avanzato", *Pharmaco Economics*, november.

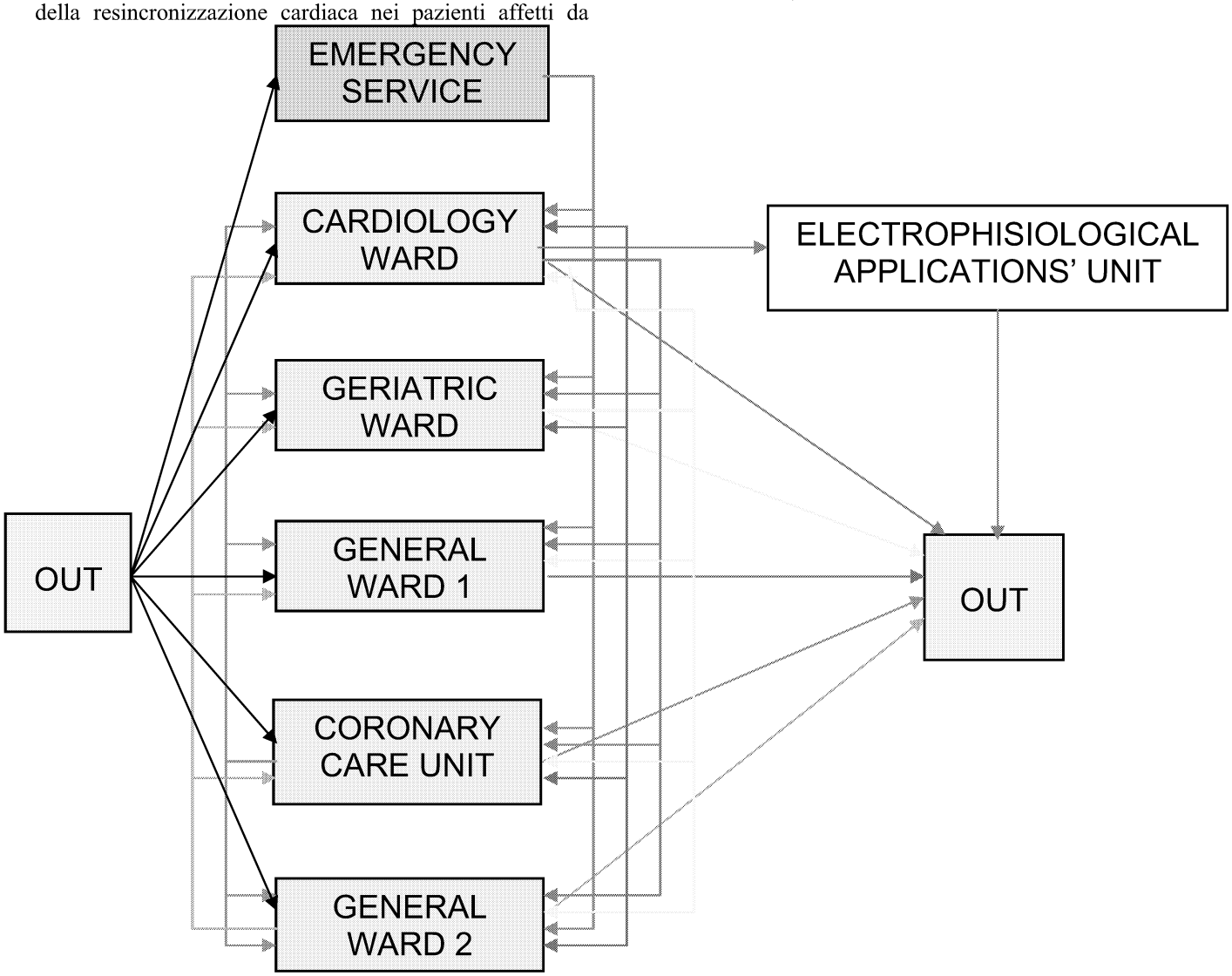


Fig. 1. Model of patient movement among hospital wards.

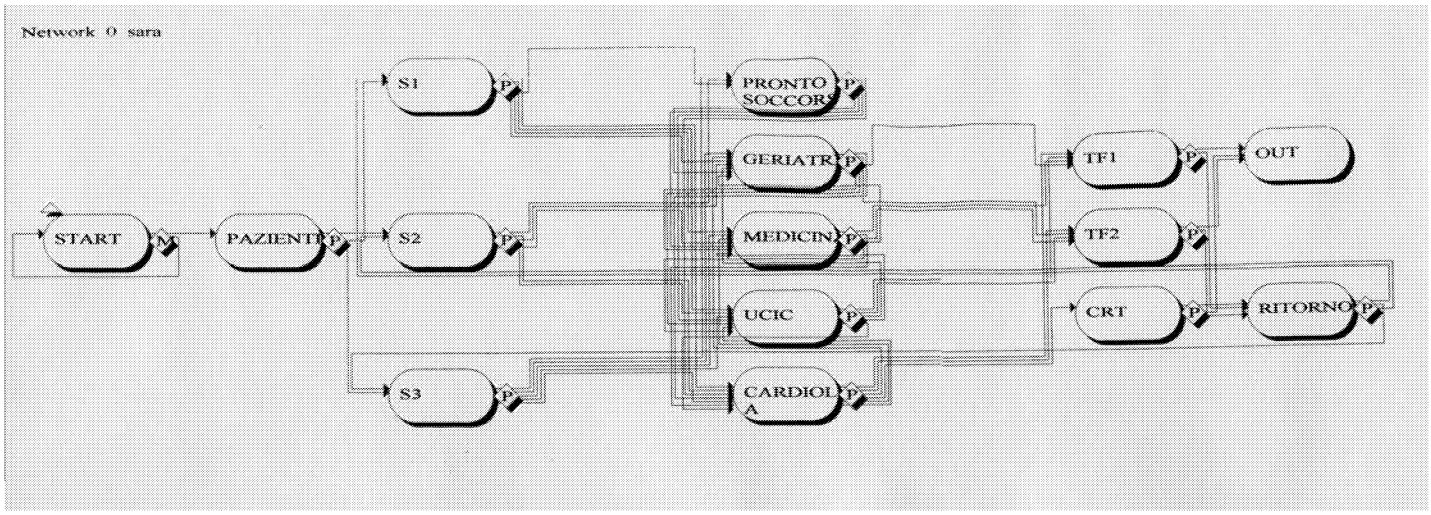


Fig. 2. MicroSaint model. Translation from Italian: PRONTO SOCCORSO = EMERGENCY; GERIATRIA=GERIATRIC WARD; MEDICINA=GENERAL WARD; UCIC=CORONARY CARE UNIT; CARDIOLOGIA=CARDIOLOGY WARD

MARINE ENVIRONMENT SIMULATION

MODELLING OF SOLUTE DISTRIBUTION IN ESTUARIES: MATHEMATICAL TECHNIQUES & COMPUTER IMPLEMENTATION

Naghmeh Keshavarzi-Roonizi
Rahim Ghorashi
Vahid Nassehi
Department of Chemical Engineering
Loughborough University
Loughborough, LE11 3TU, UK
Email: n.keshavarzi-roonizi@lboro.ac.uk

KEYWORDS
Modelling, Estuaries, Multi-mode, Computer Implementation, Solute Transport

ABSTRACT

Water quality monitoring and management in tidal waterways are facilitated considerably by the use of accurate predictive hydrodynamic and pollutant transport models. Development and application of realistic three dimensional numerical models to natural water systems however, require prohibitive amounts of hydrographic data and their use in practice is restricted by very high computational costs and effort. To overcome these problems simplified depth or cross sectionally averaged one or two dimensional models are commonly used to simulate flow and mass transport in tidal estuaries. The approximate nature of these models means that the quality of the results obtained by them can be varied and in cases where significant three dimensional effects are present they cannot be reliably used. In this paper we outline the development and computer implementation of a new approach for modelling solute transport in estuaries based on the representation of tidal hydrodynamics and solute mixing in terms of multi-mode velocity and concentration fields.

The vertical components of the filed variables in the flow and transport equations corresponding to multi-mode system are solved separately from the lateral components and thus such a model can yield information in all three space dimensions while economising computational costs and hydrographic data requirements.

INTRODUCTION

In natural water systems water quality is affected by the interactions of a wide range of hydrodynamic and physico-chemical phenomena. Therefore computer models developed to simulate hydro-environmental conditions in these systems use sophisticated techniques and architectures. In particular, in systems such as estuaries and coastal zones, where crude, large scale averaging is not acceptable, the modelling is often based on complex mathematical procedures which can preserve accuracy of the simulations. However, the necessity of maintaining computing economy as well as accuracy requires the use of modern IT in conjunction with reliable mathematical techniques. In this paper we outline a novel mathematical method which can generate reliable simulations for the transport of solute pollutants in tidal waterways. This model avoids using full three dimensional computations and hence is very cost effective. We explain that the most optimum way of implementing the present model is via the use of an IT system which provides a flexible and user friendly environment. We further describe the basic architecture of the IT system developed and provide an example of the application of the model.

MATHEMATHICAL MODEL

The derivation of multi-modal models for tidal water ways is based on the spectral (i.e. eigen function) expansion of general three dimensional hydrodynamic and solute transport equations. In general , the velocity or concentration modes satisfy Sturm-Liouville type equations. For example, the velocity modes relate to the following equation

$$\frac{\mathrm{d}}{\mathrm{d}\sigma}\left(\hat{\nu}\frac{\mathrm{d}\Phi^{(m)}}{\mathrm{d}\sigma}\right)+\mu^{(m)}\Phi^{(m)}=0, \qquad (1)$$

with $\Phi^{(m)}=0$ on $\sigma=0$ (estuary bed) and $\hat{\nu}\frac{\mathrm{d}\Phi^{(m)}}{\mathrm{d}\sigma}=0$ on $\sigma=1$ (water surface), where $\Phi^{(m)}$ are the velocity modes and $\mu^{(m)}$ are the associated eigenvalues, as indicated σ is a normalised vertical co-ordinate direction $(0\geq\sigma\leq1)$, $\hat{\nu}$ is a dimensionless shape function characterising eddy viscosity profile of flow in the tidal water way. The expansion of the velocity components, for example, in the longitudinal (i.e. x) direction are hence posed as

$$u=\sum_{m-0}^{\infty}u^{(m)}\,(x,y,t)\,\Phi^{(m)}(\sigma) \qquad (2)$$

etc. Similarly, replacing the eddy viscosity term with an appropriate eddy diffusivity, the concentration modes of solutes can be represented as

$$C=\sum_{m-0}^{\infty}C^{(m)}\,(x,y,t)\;\Psi^{(m)}(\sigma) \qquad (3)$$

Substitution of the above expansions into the general hydrodynamic and transport equations yields the modal equations used in the present work. In this model each mode represents a cross section of the water way at a certain depth and therefore solution of a number of two dimensional sets of governing equations provides the required vertical profile of the velocity and concentration fields. Detailed derivation of the multi-modal equations for tidal water ways is available in the published literature and is not repeated here (Smith, 1995). However, in the present work instead of exclusively relying on situations where an analytical solution for equation (1) is available, the general case where this equation is solved numerically is considered. This solution is based on the utilisation of a mesh independent diffuse approximation technique (Mokhtarzadeh, 1998). The expanded cross sectional equations, on the other hand, are solved using a Lagrange-Galerkin scheme whose details have been reported previously (Nassehi and Kafai, 1999).

IMPLEMENTATION

To achieve the goals of flexibility, practicality, speed, multi-user capability and computing economy the use of the described model is based on its implementation via an Information Processing Tool. The details of this system was presented in a recent international conference on Industrial Simulation (N.Keshavarzi-Roonizi *et al*, 2004). The following is a brief description of the system within the context of the present application.
The IT tool used here is constructed as a combined system consisting of three distinct modules.

- Front-end, provides a network node for communication between the users and the software. Quantitative results generated by the model are treated in a back-end module and the final result is provided in clearly tabulated or graphical formats. A log is kept by the system which records the particulars of the application for future comparisons and evaluation of the efficiency of the solution generated for a given problem. The user is guided by the front end towards inserting appropriate input data. In view of the complexity of the present application the importance of this stage cannot be over emphasized. In this application the user should be guided, firstly, to insert hydrographic data consisting of water way geometry and bed profile, and time dependent fresh water and tidal flows into the flow channel as well as bed friction, wind stress and other hydraulic parameters. These data change continuously in a moving boundary domain according to flood and ebb tides and hence the system should track and up-date the flow domain. To avoid extra complications, as mentioned earlier, we have adopted a mesh independent technique to solve the mode generator equations for velocity and concentration and the solution of the expanded equations is based on a Lagrange-Galerkin approach. The front end facilitates data handling whilst monitoring the suitability and relevance of the data inserted by the users in using the data generator (simulation) software.

- Data-generator, is a systematic library of various modelling software that use FORTRAN and symbolic programming as well as genetic algorithms to solve the model equations. Specific applications such as optimization of the bed friction values (i.e. model calibration) is carried out via genetic algorithms (Passone *et al*, 2004). The front end interface feeds the input data to the modelling sections initiating numerical computations. At the end of the computations the simulation results are passed to the back-end module for further processing required before presentation of the answers.

- Back-end, provides a network node for the import and processing of the data (i.e. results, answers, evaluations) generated by the software in the data generator module. The processed results are logged and returned to a window in the front-end for the utilization by the users. It should be noted that the simulations results obtained via the data-generator mainly consist of large amounts of numerical data and hence the back–end module re-organizes the data as preformatted tables or graphical out put. The back-end module can also communicate directly with the front-end at the preliminary stages of an application (i.e. problem simulation). For example, it is designed to provide answers regarding the validity of a given input on the basis of its self-consistency or logical status. This feature is mainly utilized to guide the untrained user to obtain required information from the system with ease and efficiency. In figure 1 the general architecture and basic procedures of the present IT system is shown.

APPLICATION AND SAMPLE RESULTS

The applicability of the developed model to realistic situations is demonstrate by the modelling of salt intrusion in the Tay estuary. This is as a typical solute transport case study in a large three dimensional tidal water way where distinct vertical variations in the solute level is expected to exist casting doubt in the suitability of using a simple depth averaged model.

The Tay estuary in Scotland, UK consists of the rivers Tay and Earn, Figure 2. It has the highest fresh water discharge of any British estuary, which consequently affects the mixing between fresh and salt water over a significant length. The estuary is 54 km and the front of salt intrusion extended for 45 km while the limit of tidal motion in near Perth, 50 km from the estuary mouth.

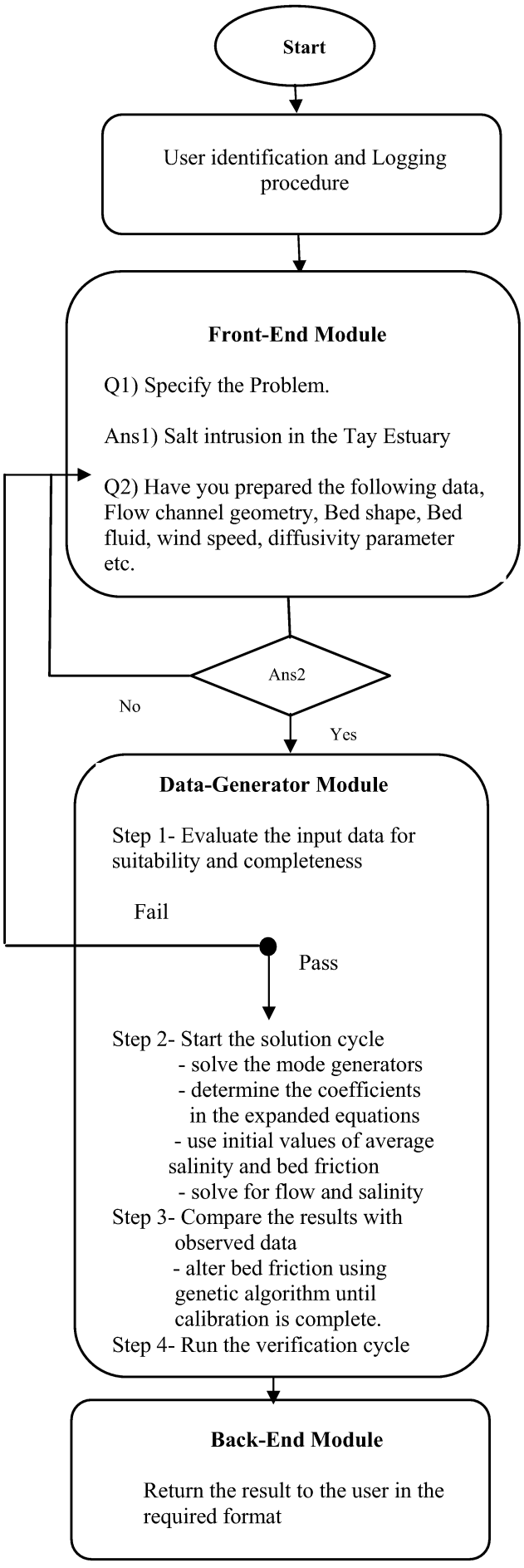


Figure 1- Schematic diagram of the IT system

The estuary is mesotidal-macrotidal with variation of the tidal range between 2.8 and 4.7 m. Hydro-dynamically influenced by its geomorphology, the Tay estuary is classified as "complex" (Buck and Davidson, 1997).

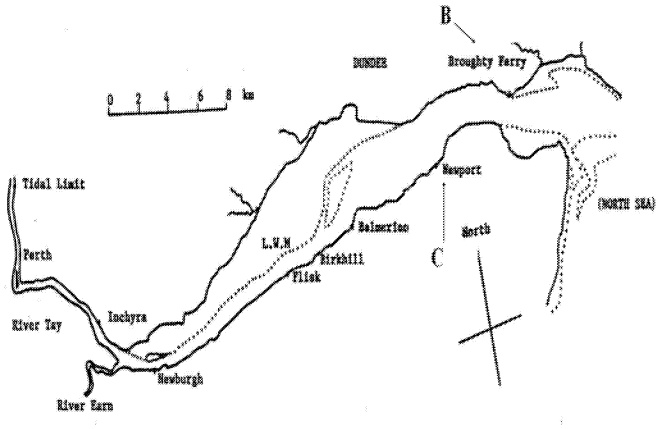


Figure2 - The Tay Estuary

The flow channel geometry in Tay has resulted from the combined effects of glaciations, river erosion and sea-level changes. Hence, the Tay Estuary is quite meandering and characterised by 165 km of shoreline. The dominant flow channel on its southern side is wide and shallow and includes a number of sand over-banks. The width and depth of estuary are not regular. From the entrance of the estuary at Buddon Ness, which is 1.9 km wide at low water, the width of the estuary increases exponentially inland to a maximum of 3 km at Dundee. At high water, the width of the estuary reaches a maximum of 5 km at Birkhill (22.5 km from the estuary mouth) with an exponential decrease from this point towards the estuary head.

Because of its typical flow behaviour and morphology, the Tay estuary has been the subject of many hydrological and statistical analyses in the last four decades (e.g. McManus, 1968) .

Considerable variation in the quantity and the velocity of water have been observed in the flow channel along the estuary. During the propagation of the tidal wave the phase lag can reach a maximum of 3 and 1.5 hours at low and high water, respectively. The propagation of the tidal wave is accompanied by the deformation in its shape, which increases with the phase lag. The irregular geometry of the estuary also affects the mixing and salt transport along the channel. While the differences in surface and bottom salinity in upper parts of the estuary are relatively small (3g/kg), salinity distributions in the vertical as well as longitudinal directions, measured at different locations in lower reaches, show significant asymmetries (Buller *et a l,* 1972).

As the brief description of the physical environment of the Tay estuary indicates the geometrical and hydro-dynamical complexity of this tidal water way is such that any accurate simulation of physical phenomenon in its environment require extensive data handling as well as elaborate and

sophisticated mathematical modelling. Therefore the simulation of salt intrusion , in particular in lower reaches of the estuary where significant longitudinal and vertical variations in salinity levels are expected to exist, provides a very appropriate case study to demonstrate the applicability of the present model and IT system. An additional factor influencing this selection is the possibility of using accurately measured salinity diffusivity data for spring and neap tides based on extensive 1972 field survey of Tay estuary (Nassehi and Williams, 1987).

As mentioned earlier after the verification of the suitability of the input data inserted by the user the front end passes the data to the simulator module and computations begin. The simulation stage starts with the solution of mode generator equations (i.e. equation 1 for the velocity and a similar equation in terms of diffusivity for salt concentration). Generated modes are used to obtain modal expansions of the field unknowns via equations 2 and 3. In practice, at this stage the values of coefficients in the multi-modal forms of the hydro-dynamic and salt transport models which have been written in a FORTRAN code are determined. Therefore after this stage the multi-mode equations can be solved by the finite element method to generate the field variables.

The hydro-dynamic equations for velocity components are initially solved assuming an average salinity throughout the estuary and inserting a pre-determined set of bed friction values (i.e. Manning's coefficients). The initial determination of the bed friction coefficient is based on the types of sediments in the flow channel bottom and bed morphology. After the initial solution of the flow equations the salt transport equation is solved. The cycle is then iterated by updating the salinity and friction coefficients so that flow equations yield results which are close to a selected set of observed data for a given tidal event. This is step is commonly known as the calibration step in the simulation of tidal phenomena (Nassehi and Bikangaga, 1993). The adjustment of bed friction is carried out automatically via a genetic algorithm (Passone, 2002) which significantly speeds up the model calibration. After the calibration the model is verified by running it for a different tidal event whilst all of the physical parameters are kept similar to the values determined by the calibration and not altered.

The simulation vertical salinity profiles for the calibration event (Spring tide of 12 June, 1972) at stations B and C (Figure 2, corresponding to Broughty Ferry and Newport along the flow channel) are shown in figure 3. AS this figure shows significant variations in the salt concentration throughout the tidal cycle in the water column at these locations are predicted. These profiles are compared with depth average salinities to provide a quantitative measure for the deviation from reality that will result if the salinity changes in the water column is ignored.

The results obtained for the verification event (i.e. neap tide of 20 June, 1972) at stations B and C are shown in figure 4. Despite the significant reduction in water level changes during the neap tide noticeable variations in the vertical salinity profile throughout the tidal cycle is again predicted. The predicted asymmetry in the salinity profiles between stations B and C and also with respect to time are as expected for a complex estuary such as the Tay.

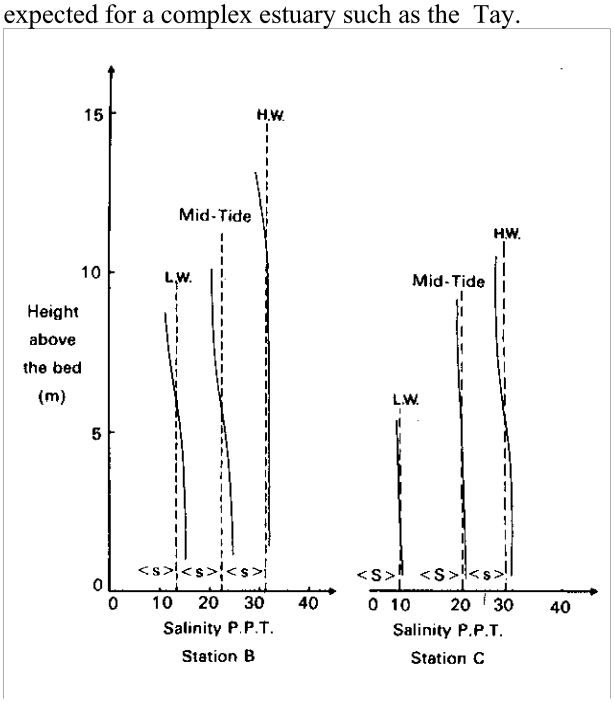


Figure 3- Salinity Profiles, Spring tide of 12 June 1972

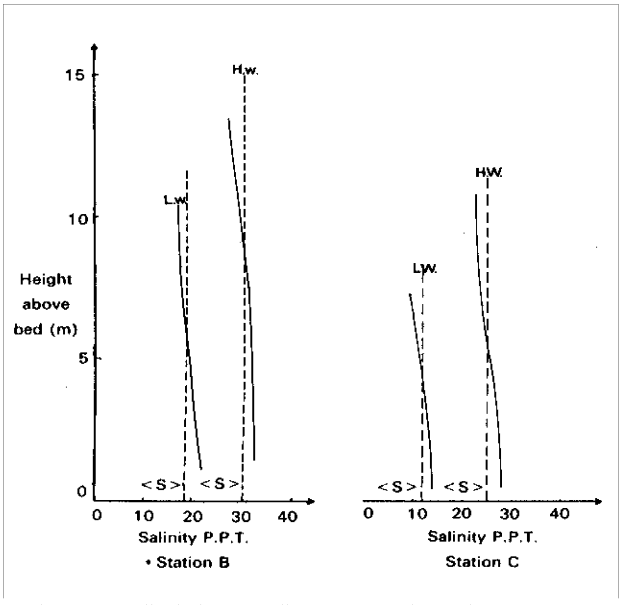


Figure 4- Salinity Profiles, Neap tide of 20 June 1972

CONCLUSIONS

The novel modelling approach presented in this paper is shown to yield accurate solute concentration profiles in a water column (i.e. bed to surface direction) in tidal water ways. In this approach full three dimensional computations are avoided and hence the model provides a very cost effective scheme for obtaining results for complex tidal water ways under realistic conditions. To overcome the

complexity of the implementation of the developed multi-stage software and to make it accessible to users who may have a minimal knowledge of mathematical modelling the scheme should be used in conjunction with an information processing tool. The basic structure of the proposed IT system is explained and its application to a realistic case study is presented. The results obtained by the model for the salinity profiles in lower reaches of the Tay estuary reveal detailed information regarding solute concentration variations during a tidal cycle in the water column at selected locations along the estuary. Critical evaluation of the generated results show their theoretical consistency and close comparison with realistically expected data. The developed model is a general tool and can be applied to simulate a variety of hydro-environmental phenomena in complex estuaries and coastal zones.

ACKNOWLEDGMENT

The authors wish to thank Engineering and Physical Research Council of UK (EPSRC) and Loughborough University for PhD studentships which enabled the development of the model and information processing system described in this work.

REFERENCES

Buck A.L, Davidson A.C., 1997, " An Inventory of UK Estuaries", Joint Natural Conservation Committee, Peterborough

Buller A.T, Charlton J. and McManus J., 1972, "Data from Physical and Chemical Measurements in the Tay Estuary for Neap and Spring Tides" Tay Estuary Research Report *No.:2,* University of Dundee.

Keshavarzi-Roonizi N., Ghorashi R. and Nassehi V., 2004, "An Information Processing Tool for the Management of Natural Water Systems", *Proceeding of the 2nd International Conference on Industrial Simulation, ISC 2004* , Malaga, June 7-9, pp 40-44.

McManus J. 1968, " The hydrology of the Tay basin in Dundee and district ", (Ed. Jones,SD) *British Association for the Advancements of Science, Dundee,* pp 107-124

Mokhtarzadeh M., 2002, "A General Global Approximation Method for the Solution of Boundary Value Problems", PhD thesis, Loughborough University.

Nassehi V. and Williams D.J.A, 1987, "A mathematical model for salt intrusion in the Tay Estuary", *Proceedings of the Royal Society of Edinburgh,* 92B, pp 285-295.

Nassehi V. and Bikangaga J .H., 1993, "A mathematical model for the Hydrodynamics and Pollutants Transport in Long and Narrow Tidal Rivers", Applied Mathematical Modelling, 17, pp 415-423.

Passone S., 2002, "A hybrid reasoning system for supporting the modelling of estuaries", PhD thesis, Loughborough University.

Passone S., Nassehi V. and Chung P.W.H., 2004, "A Hybrid Reasoning System For Supporting Estuary Modelling", *Journal of hydroinformatic (in print)*

Smith R., 1995, "Multi-mode models of flow and of solute dispersion in shallow water: Part 1. General derivation", *Journal of Fluid Mechanics*, vol.283, pp. 231-248.

Methodology for Holarchic Ecosystem Model based on Ontological tool

Guillaume Prévost, Pierrick Tranouez, Sylvain Lerebourg, Cyrille Bertelle and Damien Olivier
LIH - Université du Havre
25 rue Ph. Lebon - BP 540
76058 Le Havre Cedex - France
E-mail : guillaume.prevost@univ-lehavre.fr

Keywords

Aquatic Ecosystem, Food Chain, Individual-Based Model, Holarchic systems.

Abstract

In this paper, we present a methodology for aquatic ecosystem modeling. We based our model on the holarchic (non directional hierarchic) nature of the ecosystems. Aquatic ecosystems are crossed by structuring fluxes (light radiation, mass transport, etc.) which confer multi-level of emergent organizations. Elsewhere, a part of aquatic ecosystems are species organizations called food chain. These ones are themselves interacting complex subsystems with multi non-directional retro-actions between them and their environment. These retroactions can be modeled at different levels. Thus, we propose an hybrid holarchic compartmental model. This one aims at easing the gathering of phenomenon in a multi-model, multi-level simulation for studying food chain. We present a methodology based on the use of an ontological tool, called Protegé 2000 and allowing an interface with biological specialists to describe animal behaviors with a hierarchical and structured approach. Elsewhere, the ontology is also based on accurate description of all the species of emergent organizations (homogeneous structures or complex functional systems) that we have to manage dynamically during the simulation. The implementation is build on ProActive package which manages dynamic distribution of simulations over networks, using code migration, for example.Results of first simulations using this environment are shown. We so propose a constructive methodology for concrete simulations over distributed computations, allowing to study real problems in all their complexity, specifically in multi-scale description.

1 ECOSYSTEMS ARE COMPLEX

The botanist Tansley (Tansley, 1935) defines ecosystems as "The more fundamental conception is ... the whole system (in the sense of physics) including not only the organism-complex but also the whole complex of physical factors forming what we call the environment. We cannot separate them (the organisms) from their special environment with which they form one physical system ... It is the system so formed which [provides] the basic units of nature on the face of the earth ... These ecosystems, as we may call them, are of the most various kind and sizes."

Typically, ecosystems are described as a biotope and a biocenisis in mutual interaction (S.E.Jorgensen and Müller, 2000). Moreover, they are crossed by fluxes (mass transport or energy) which dynamically structure them. A reductionist approach fails in modeling such mutual interactions and feed-back processes. New approaches based on general system theory concepts have therefore been tried to produce more efficient models.

Ecosystems are systems as described in the general system theory (Le Moigne, 1994) and can be seen as a set of interacting elements which are characterized by following aspects:

- **mutual dependence.** Each element is directly linked with other elements in structure or dynamic. Therefore, its evolution depends on the other elements in interaction with it. Finally, separating an element from its neighborhood modifies it.
- **emergence of organizations.** The interaction of elements leads to the emergence of natural organizations which generate "new entities". Those entities differ from their components in their structure and dynamic.
- **feed-back processes.** This is the retroaction from the natural organizations to its own components.

Emergence processes act recursively and generate hierarchical systems organization. An adapted description is the one called SOHOS according to Koestler (Koestler and Smythies, 1969). SOHOS stands for Self-Organized Holarchic Open Systems. An holarchy is a non-directional hierarchy in which the members are called holons.

If we consider the three previous characteristic aspects of ecosystems, an adapted definition of ecosystems could be: "biotope+biocenosis", natural multi-level holarchic systems, crossed by structuring fluxes.

Ecosystems as holarchic systems, must be studied at many levels of time and space (each level is significant) and are crossed by fluxes. Many tools exist to model and simulate ecosystems. Those tools differ by the nature of the model they use, the level they describe and the phenomena they directly take into account. Thus, they mainly focus on one aspect of the ecosystem. Therefore, gathering all these models is a hard task due to structural differences between these tools. Mixing these models is seldom theoretically feasible, and when it is, the computation of the resulting simulation is often crippled by a terrible computation time. Unfortunately, raising the number of interactions modeled is the concern of most modelers. We aim at providing a model which would facilitate the gathering of models for simulating aquatic ecosystems in a multi-scale description.

2 AN HOLARCHIC HYBRID MODEL FOR ECOSYSTEMS

First, let's start with a description of an ecosystem life cycle. An ecosystem always go through three states characterized by its biotope and its inner complexity.

- To begin with, the profile of an ecosystem can be described as juvenile. The ecosystem itself contains many raw materials and its biodiversity is low. The biotope is mainly made of simple organisms. Those organisms modify the layout as they keep on multiplying and consuming the raw material. The environmental condition has little influence on their development. The main factor limiting their growth in size and number is the quantity of material and space.

- Then, an ecosystem evolves into a mature or adult form. An adult ecosystem undergoes a replacement of its original settlement (simple organisms) with complex organisms. Those organisms suit the ecosystem characteristics and tend to maintain them. They consume and produce (or participate to the production of) raw materials. Those organisms replace the simple organisms (but some simple organisms remain). So on, the biodiversity of the ecosystem is very high. We should be aware that the complex organisms need the building made by the simple organisms during the juvenile stage to appear.

- Finally, an ecosystem tends to be aging. When being adult, the ecosystem is made of many types of complex systems, each one participating to the maintenance of the system. As it grows old, the ecosystem tends to lower his complexity and biodiversity by eliminating the less performing complex organisms and conserving the best association of organisms.

During its life, an ecosystem always undergoes stress period making it pass from one step to another (sometimes regressing as it is the case when an ecosystem is being exploited). We point out that biotope and biocenosis are directly linked and that each organism acts for his ecosystem as the same time as it is influenced by it.

Following this line of thought, we adopt a well-known description of the biotope based on a classification of the organisms. That classification is made of three parts:

1. **Producers.** Producers are the base of the ecosystem. They are responsible for the production of organic mass by consuming inorganic materials (mineral salt for example). Moreover, they produce many substances and heat. They too are eaten by consumers.

2. **Consumers.** An ecosystem most of the time is studied through its organic mass. In an ecosystem, most organisms try to develop and support its organic mass. That category of organism is consumers. They mostly produce organic mass by consuming other organic mass and release part of that mass in the ecosystem. They too participate to the layout of the environment .

3. **Detritivors or decomposers.** As an organism lives, it releases (dead) organic mass. That organic mass is reused by bacterias to produce raw materials needed by the producers. They thus maintain the ecosystem resources.

Each part of that classification is a part of the functional aspect of the ecosystem and participate to the previous cycle of life. That classification constitutes the base of the representation of the biotope in our model.

We represent ecosystems with a multi-level model. Some level are fixed "a priori". We shall describe them later. The determination of the level lays on the following concepts.

Ecosystems are systems as described in the general system theory (Bertalanffy, 1968) meaning they are made of elements in interaction. In ecology, the basic elements are individuals. So on, we should introduce individuals in our model. Clearly, we should use an individual based model on that level. Moreover, ecosystems are thermodynamical systems. We should take into account flows between ecosystems on a different level than the individual's one (as they operate on a different scale). Finally, those flows are well modeled by laws.Finally, ecosystems are SOHOS. That induces we shall define non-directional relations between levels and furthermore, and add dynamic scaling for emerging entities.

While studying the informations available concerning several natural aquatic ecosystems (the Seine estuary, for example) we noticed that these ecosystems could be separated into different compartments, each one being a particular ecosystem. Clearly, that description is space-oriented depending

of the localization of the compartment and its inner space. A level corresponding to those compartment should be adopted.

Definition of the holarchic hybrid model

Following the previous concepts, we propose a model suiting ecosystems features and allowing reuse of already existing model different by nature (law or rule-based, different scales).

The model is individual-based in the way it represents the different entities of an ecosystem. As a consequence, each entity has states (e-states, i-states or p-states) and behaviors. The behaviors correspond to the model used to model the dynamic of the entity. It could be rule-based or law-based.

Our model presents three levels defined "a priori".

- **Individual level.** That level is the lower one in the holarchy of our system.

  It embodies entities that could not be decomposed. We introduce individuals and super-individuals at that level. Moreover, the elements of that level are described following the consumer-producer-detritivor model. Thus, one should question what are the links between a particular entity and the ecosystem. That level is clearly individual-based so behaviors of the entities are rule-based and entities have i-states.

- **Compartment level.** The space is a critical data in ecosystem modeling. We introduce the space at that level . Thus, a compartment is a single entity in interaction with other compartment (exchanging flows) and containing individuals. As ecosystems are SOHOS, individuals and compartments influence each other. The compartments have law-based behaviors and e-states. The e-states of the compartment correspond to global values considered homogeneous in all the space of the compartment.

- **System level.** Systems are non-spacialized entity corresponding to a set of entity. The link between those entities could be defined "a priori" or during the simulation. The first example of a system is the ecosystem itself. It is made of compartments and defines their relations. Moreover, phenomena occur at the scale of the ecosystem itself influencing all the compartments. The systems have e-states and law-based behaviors. During the simulation, different kind of systems could emerge, each one corresponding to a new scale. Moreover, phenomena with particular scales should be modeled through systems. It is important to understand that the phenomena occurring on the ecosystems level directly modify the compartment and thus influence too the individuals of the compartment themselves. Simultaneously, individuals modify their compartment and thus influence the ecosystem.

What will be modeled

We tested our model on a classic example of aquatic ecosystem. We will now describe it.

Our example is an ecosystem in which we study the influence of light and oxygen on a simple food chain. The light differs in many place of the ecosystem. The oxygen influences the behavior of the biotope.

The food chain is made of four species. The first population is made of planktons consuming mineral salt and producing oxygen depending on the light. Bacterias constitute the second one. They decompose organic mass and release mineral salt. The two last populations are fishes. The first of them feeds on planktons. The second have plankton and the first population of fish as preys. Both of them consume oxygen.

3 AN ONTOLOGY TO DESCRIBE ECOSYSTEMS

As we propose a model, we also provide a tool to ease the modeling of ecosystems using our work. That tool comes as an ontology of our model.

What is an ontology

First, let's remind what's interesting in making an ontology. 'Ontology' stands for a formal definition of the terms and concepts of a domain. While making an ontology, we start by enumerating the crucial concepts of the domain. Then, we should give a clear definition of each one.

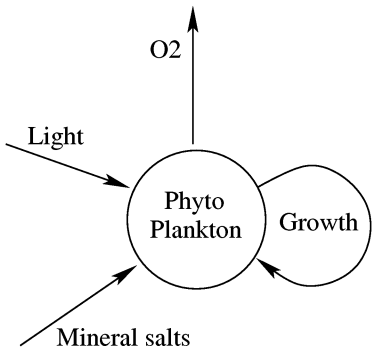


Figure 1: Producer - phytoplancton

Finally, we should obtain a collection of concepts. Therefore, links and relations between those concepts have to be established. We can distinguish different types of links. A concept could be a precision of another one. In that case, it is said that one of them "inherits" from the other. Concepts could have relation when a concept "has something to do" with another. That formal link determines what other

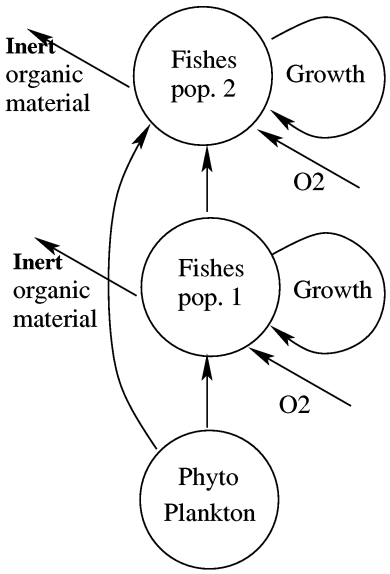


Figure 2: Consumer - food chain

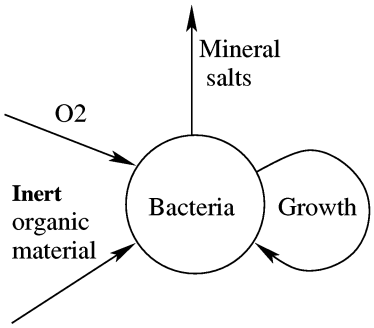


Figure 3: Decomposer - bacteria

concepts should be considered when you think about one of the concepts.

Finally, ontology precises the view of a domain, gives an explicit definition for each concept and highlights the assumption of our view. We use Protege2000 (Protege, 2000) to design our ontology. Understanding and following the assumption of our model is natural while using that ontology to study a particular case of ecosystem.

Our ontology applied to the example

The following demonstration correspond to a down-top approach, but we can also use a top-down one to model our ecosystem. First, the example shows us a food chain composed of four species. As we want to introduce the species, we should first determine if they are consumers, detritivors or producers.

- **Plankton.** Plankton produces oxygen, consume mineral salt and multiply without consuming organic mass. it clearly corresponds to producers.
- **Bacterias.** Bacterias use the organic mass to produce salt through decomposition. They are detritivors.
- **Fishes.** As far as we know, fishes are consumers as they consume organic mass to produce their own.

As we define those species, we see that producers, consumers and detritivors must be linked to a particular individual-based model (individual or super-individual). We should now think about the way we will represent them in our system. As we made that choice (individual for the plankton and fishes, super-individual for the bacterias), we must define the behaviors of our individual. Each category has already defined behaviors that can be reused. We can also introduce a new rule-based model that will be used by the behaviors to simulate the photosynthesis for example. Simultaneously, the ontology asks us which way the individuals will be localized in the space. Therefore, we should look for a localization suiting our needs. It will also define the way the space will be modeled. We choose a 2D coordinate localization. The ontology induces that we should choose a space corresponding to that kind of localization. We then choose a 2D continuous space.

The lowest level of our simulation has been achieved. Fortunately, new questions has been revealed by our ontology. Mineral salt, oxygen, light and organic mass needed by the low-level entities have to be modeled. First, should we introduce them as producers, consumers or detritivors? The best way to reintroduce them is placing them at the compartment level as they correspond to e-states. The light is a criterion to distinguish the different compartment. The mineral salt, oxygen and organic mass quantities vary in a compartment depending of its settlement. Therefore we define three e-states. Their behaviors are simple. Oxygen,

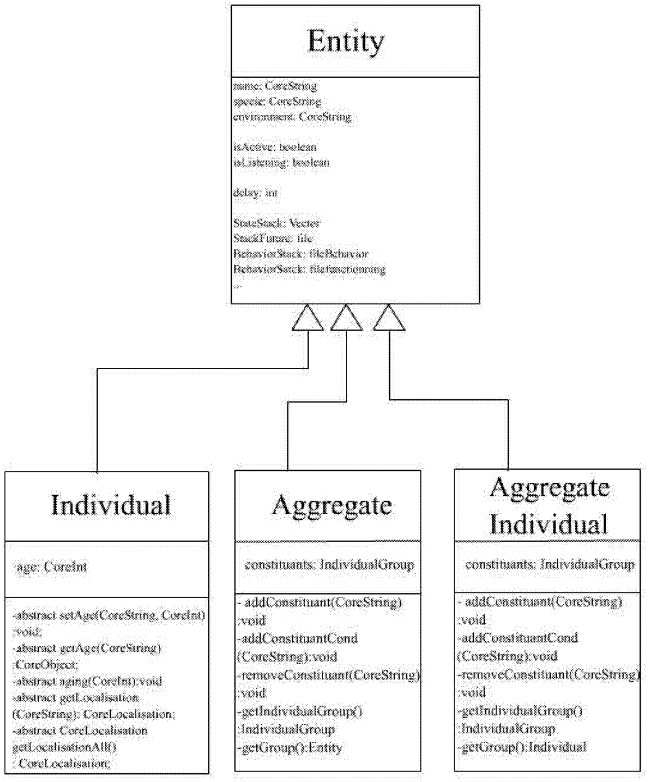


Figure 4: An example of Class:the class Entity and its child

mineral salt and organic mass don't vary by themselves nor have subtle interactions with other elements of the simulation so they have no behavior. Light could vary depending on the time so we can add a behavior making it lower and higher (actually, we won't).

During the previous step, the ontology led us through the definition of our compartments. They will have four e-states (O^2, mineral salt, light and organic mass). The ontology asks us the types of space model used with that compartment. We have already chosen a 2D continuous space one. It only remains to define the behaviors of our compartment. A compartment exchanges fluxes with its neighbors. So we should define a law-based model corresponding to the fluxes between the different compartments. When defining it, we now realize that we must precise how compartments are linked. Therefore, compartment have a neighborhood and values determining how much they are connected to each neighbor. A compartment has also a localization corresponding to the space model "neighborhood".

The informations concerning the neighborhood appear at the level of the ecosystem itself. That one constitutes the higher level of our simulation and embodies the compartments. It also defines the neighborhood as its own space. Its behaviors correspond to the global exchange of the ecosystem with the "outer world" (classical notion in ecology).

To conclude with, the ontology naturally guides us through the modeling of a simple example of ecosystem respecting the concepts and assumptions of our own model. That step ends with the choice of classical law-based or rule-based models to simulate the behaviors of each entity. That modeling step achieved, we can now go to the simulation step.

4 FROM MODEL TO SIMULATION: A DISTRIBUTED IMPLEMENTATION

The previous work provides us with a model corresponding to our ecosystem and the aspects we want to study, but the purpose of this model is to give birth to a simulation, allowing us to increase our understanding of the ecosystem. Therefore, we are conceiving a platform corresponding to our model and directly linked with our ontology.

Simulation framework description

First, let's sketch that the framework will follow the concepts of our ontology. We will based the framework on active objects as they suit well the individual-based model chosen in our hybrid model. To easily conceive a simulation tool using active objects, we program in java using ProActive.

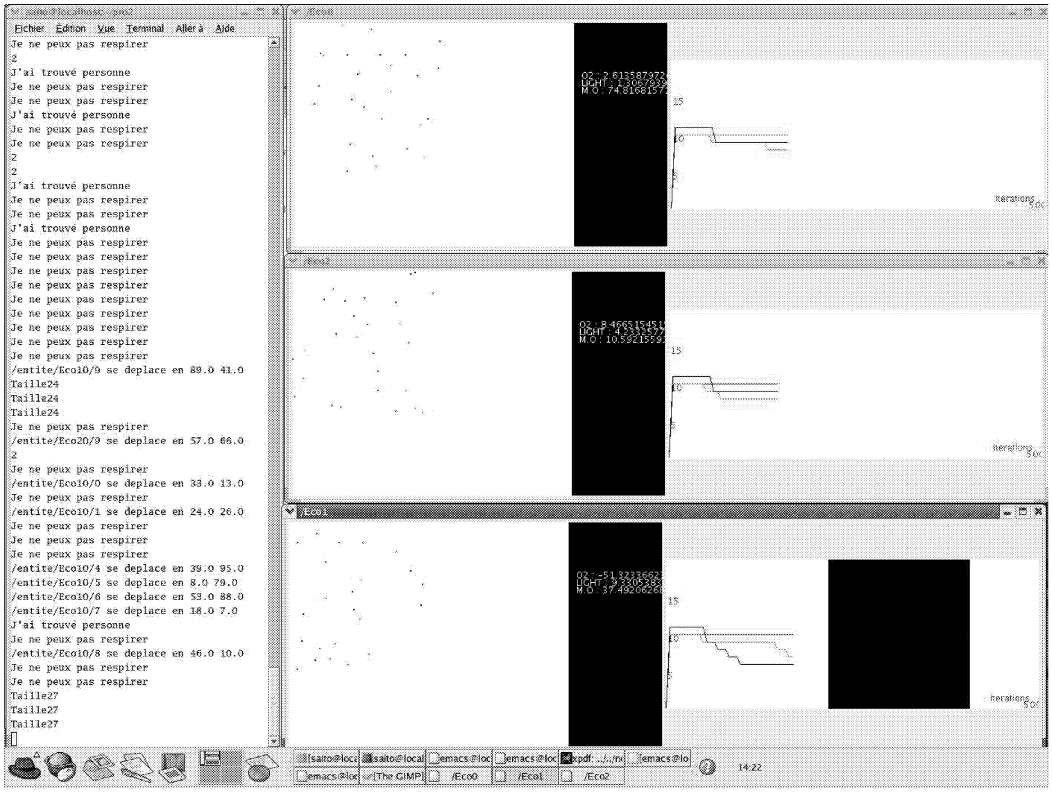


Figure 5: Screenshot of a simulation

ProActive (Baude et al., 2000) is an API enabling the conception of distributed active object's applications. ProActive is made as a library of static java classes. The main class (ProActive) allows us to turn Java Object into ProActive Objects (as long as the original object is correctly define for ProActive). Therefore, ProActive Objects are registered among the network and can communicate asynchronously or migrate. The migration conserve both the object's state and its execution state. Remote Method Call on distant objects are non-blocking. Therefore, we implement all the concepts of our model in a reusable way.

Our framework lays on several classes:

- **Entity.** Every object acting during our simulation is an entity (i.e an active object). It has states, behaviors and a fixed cycle of life. Ideally, each entity have its own execution thread (i.e entities are ProActive Active Objects). To maintain a satisfying simulation's speed, special execution objects gather a fixed number of entities and manage their functioning. Even so, entities are active objects and have their own single name (i.e URI) known by all the computers of the network. The choice of gathering entities or not is free and the numbers of entities within an execution object as the number of execution objects on a computer can be fixed by the modeler. This one don't have to manipulate execution objects and only have to define entities.

- **Aggregate (i.e systems).** Aggregates are entity containing other entities. They manage the individuals suppressing dead ones, assuming migration of objects in the network.

- **Compartment.** Compartments are aggregates defining a space and making part of a neighborhood. Compartments have states (e-state or p-state). The states are implemented as Characteristic values which can be active (with their own behavior) or not. In the last case, the compartment's behaviors make them evolve. Compartment behavior's also include flux exchange with other compartments of its neighborhood.

- **Individuals.** Individuals are entities localized in a compartment that cannot be broken down. We distinguish three types of individuals: single individual, super individuals and possibly e-state with behaviors. The individuals have a localization representing its placement within its compartment space. The type (or class and model) of the localization depends on the individual nature.

- **Behavior.** A behavior is a task done by an entity during its life-cycle. Behaviors correspond to the computing application of a model onto an entity. Behaviors can be executed one time, or in a cyclic way. A generic behavior consists in a cyclic call to a method. For example,

photosynthesis could be define as a method in a phytoplankton class. The corresponding behavior would regularly invoke that method depending on the object states and on its local environment. Behaviors corresponding to automata are also usable.

- **Space.** A class representing the main characteristic of spacial representation in our simulation. For now, two types of space are available. The grid one consists in a traditional 2D grid. A continuous space is also proposed. The system permits the use of multiple space at the same time on condition that one can pass from a space representation to another. Therefore, one can use multiple type of model (individuals, super-individuals). Moreover, grid induces introduction of a local memory in the space useful to speed up the simulation. Continuous space solve some problems due to local heterogeneity.

- **Localization.** A class representing the main characteristics of localization of an entity in a space. Super Individuals can be localized with:

  - coordinates corresponding to a grid
  - coordinates and a radius defining a circle area corresponding to the Super Individual on a continuous space.

  Individuals are localized with coordinates. As Individuals and Super-Individuals are involved at the same time in simulations, a conversion system makes the different type of localizations compatible. The localization library will be expended.

- **Living Part.** Those classes represent the main informations needed to incorporate a consumer, a detritivor or a producer in a simulation (inducing the model that correspond to the photosynthesis, decomposition ...). Therefore, when creating an individual or a super-individual, the modeler should define methods the behaviors would call during the simulation.

Many other classes have been developed in addition to those ones that are purely computational, with no meaning in the ontology. Our goal is to provide an easy way to reuse and complete framework. Moreover, the final platform will be completed with an interface guiding ecologists through the conception of a simulation without coding.

Framework test on our simulation

The test of our framework on the simple example previously developed highlights some aspects of our work. First, developing a simulation respecting the model tends to be easy as little programming needs to be added. Hopefully, the classes and functions already developed fulfill all the operative aspect of the simulation.

Despite that fact, there is a blatant need to incorporate existing models to make a behavior library useful for ecologists.

The simulation corresponding to our example shows interesting results. In some case and depending on the neighborhood constitution and on the light ratio, compartments tend to eliminate some species. Thus, some compartments specialize in producing oxygen and are full of plankton and bacterias. In that case, the first population of fish only feed on the other fish and live in a compartment deprived of planktons. In other case, the three compartments tend to a moderate equilibrium. In that case, the species appear in every compartment corresponding to a certain ratio of individuals for each one of the species.

5 CONCLUSION

Some aspects of complex organizations in aquatic ecosystem, can be modeled according to the approach previously presented.

The next step of our work is thus to confront in-situ experimental behaviors with some computer based behaviors obtained with the proposed simulation. Using more complex descriptions for the biodiversity, we will be able to produce some explicative and predictive modelizations for auto-organization phenomena in biological systems.

To realize these perspectives, we have already obtained many scientific informations on the Seine estuary which is a very complex ecosystem in the north of France. Last years, many studies have been made to give precious informations for understanding the complexity of this natural environment (Guézennec et al., 1999). Next years of our studies will concern the development of concerted models taking into account information from specialists of the Seine estuary ecologist .

Thus we will introduce in our simulation copepods which dominate the zooplankton. They are among the most abundant metazoans on earth and can be found almost everywhere there is water - from high-altitude lakes in mountain ranges, down to the depths of the ocean trenches. They are super-abundant in sediments, and they are parasitic on virtually every phylum of animals from sponges to chordates (including reptiles and mammals). Our future work will considered the key processes when copepods are modeled at the individual scale and a particular attention will be paid to feeding processes (encounter, attack, capture, ingestion) and also to physiological processes (development, metabolism,reproduction, mortality, . . .).

Grants

G. Prevost thanks the "Région Haute-Normandie" for its support concerning his contribution to this study.

References

Baude, F., Caromel, D., Huet, F., and Vayssier, J. (2000). Communicating mobile active object in java. In *Proceedings of HPCN Europe*.

Bertalanffy, L. V. (1968). *General System Theory foundations development applications*. George Brazille Inc., New York.

Guézennec, L., Romana, L., Goujon, R., and Meyer, R. (1999). *Seine-Aval : un estuaire et ses problèmes*. Programme scientifique Seine-Aval. Ifremer editions.

Koestler, A. and Smythies, J. (1969). *Beyond reductionism*. Hutchinson.

Le Moigne, J.-L. (1994). *Modélisation des systèmes complexes*. Dunod.

Protege (2000). *The protege project*. Protege. http://protege.stanford.edu.

S.E.Jorgensen and Müller, F., editors (2000). *Handbook of Ecosystems Theories and Management*. Lewis Publishers.

Tansley, A. (1935). The use and abuse of vegetational concepts and terms. *Ecology*, 16(3):299.

MODEL OF THE NITROGEN FIXATION PROCESS IN MARINE CYANOBACTERIA

Sophie Rabouille, Karline Soetaert and Marc Staal
Netherlands Institute of Ecology – Centre for Estuarine and Marine Ecology,
PO box 140, 4401 NT Yerseke
The Netherlands.
Contact : s.rabouille@nioo.knaw.nl

KEYWORDS

N_2 fixation, cyanobacteria, *Trichodesmium*, chemostat, mathematical model.

ABSTRACT

A mathematical model is described that represents the growth and N_2 fixation process in the nonheterocystous cyanobacterium *Trichodesmium* sp. N_2 fixation supplements the common N-uptake mode of nutrition but implies extra metabolism costs. The paper focuses on the model hypotheses and describes how the interactions of different N sources are simulated. The types of temporal patterns and dynamics the model exhibits are discussed.

INTRODUCTION

N_2 represents 78 % of the earth atmosphere but constitutes a growth-limiting nutrient in most marine systems. Algae exploit the inorganic nitrogen (DIN) dissolved in the water mainly as nitrate (NO_3^-) and ammonium (NH_4^+). Selected groups of autotrophic bacteria are also able to use N_2 as an alternative N source and in certain environments diazotrophy is a non-negligible source of nitrogen into the system (Bergman et al., 1997; Capone et al., 1997). N_2-fixing cyanobacteria can greatly influence community production and can impact the coupling of C-N-P cycles, as well as the net oceanic sequestration of atmospheric carbon dioxide (Karl et al 2002). Whereas models are being widely developed at the ecosystem level to describe the global nitrogen cycle (Hood et al., 2001; Fennel et al., 2002; Moore et al., 2002), the mechanism of N_2 fixation, at the population scale, remains poorly accounted for in theoretical works. A functional approach is needed to explicitly represent the growth of diazotrophs and assess its ecological control by environmental factors.

Nitrogen fixation is an enzymatic process carried out by the nitrogenase enzyme. This activity is energy consuming, and within cyanobacteria depends on the products of photosynthesis. But as nitrogenase is highly sensitive to oxygen, cyanobacteria have evolved several strategies to separate N_2-fixation from the O_2-evolving photosynthesis (Bergman & Carpenter, 1991; Gallon, 1992; Bergman et al., 1997; Gallon 2001; Berman-Franck et al 2001). Heterocystous species have evolved specialised cells, the heterocysts, to spatially separate the nitrogenase from photosynthetic cells. In contrast, most nonheterocystous species temporally separate both processes: they actively fix nitrogen during the night (when O_2 is not produced). In this strategy, the N_2 fixation process depends on the energy reserves stored during the day.

We developed a mathematical, physiologically based model that describes the growth of nonheterocystous cyanobacteria. At the same time, laboratory experiments are conducted to estimate the processes rates. The model takes into account the interaction between the nitrogen and carbon metabolism and emphasises the control of abiotic and biotic conditions on N_2 fixation. Here we present the model hypotheses that are relevant for the N_2 fixation process. We demonstrate the types of temporal patterns the model can exhibit and highlight how N limitation and N-fixation interact in the model.

We focus on *Trichodesmium* sp., a colonial, nonheterocystous cyanobacterium of global ecological significance (Capone et al, 1997). One special feature in *Trichodesmium* is that N_2 fixation occurs concomitantly to photosynthesis. This activity shows a maximum around midday (unpublished data). Moreover, Mulholland et al. (2001) showed that nitrogenase is also affected by the N source. In particular, nitrogenase synthesis is lowered if nitrate is available in the medium, and its activity is regulated by the intracellular concentration of recycled metabolites as NH_4^+ and glutamine (an early assimilatory N-product).

Short Description Of The Model

We consider the modeled system as a well-stirred chemostat. The model is formulated in continuous time and based on a set of ordinary differential equations. It describes mass transfers in terms of carbon and nitrogen in a population, including a mechanistic description of the N_2 fixation process. In the simulations that we will discuss, phosphorus is assumed never limiting and self-shading is not simulated. The irradiance applied is a 12h/12h Light/Dark sine regime. The model is implemented on a PC in FORTRAN, in a simulation environment, FEMME (Soetaert et al., 2002).

A representation of intracellular reserves seems compulsory to accurately describe the dynamics of the N_2 fixation and its regulation by the environment. To date, different physiologically based models already describe the uncoupling of carbon and nitrogen assimilation in algae. In such unbalanced models, internal reserves consisting of either energy reserves (carbon) or intracellular nitrogen are included as an intermediate between photosynthesis and growth (Lancelot & Billen, 1985; Droop, 1973; Tett, 1998). These

formulations have proved their accuracy in models related to cultures (Geider et al., 1998) as well as to natural environments (Lancelot et al., 1991; Soetaert et al., 2001; Flynn & Fasham, 2003; Van den Meersche et al., 2004). Recently, Stephens et al. (2003) devised a model that simulates the interaction between the different nitrogen sources in the nonheterocystous diazotrophic cyanobacterium *Gloeothece*. Uncoupling of carbon and nitrogen assimilation is achieved through the description of the intracellular concentration of a nitrogen reserve (glutamine) and includes the description of nitrogenase activity. Thus the changes in the C:N ratio are driven by fluctuations of the glutamine pool (N-reserve), whilst the accumulation of storage carbon (and hence energy reserves) is not explicitly included. Because of that, the representation of the interactions leads to a complex set of equations. To make the interpretation easier, both of the analysis of the model behaviour and the description of N_2 fixation from a mechanistic point of view, we chose a simpler formulation. We adopted the decomposition of cellular carbon components proposed by Lancelot & Billen (1985) in which an intermediary pool of low molecular weight carbon molecules, the product of photosynthesis, is described in addition to the internal reserves of carbon. The pool of small carbon molecules has rapid turnover which allows representing fast fluctuations of the processes. Added to that description, we also included an internal storage of nitrogen reserves to account for the nitrogen fixation process. We thus propose a description of *Trichodesmium* colonies through a set of five state variables:

X_M (mmol C m^{-3}) is the pool of small carbon metabolites that serves as intermediate in all the processes involving carbon fluxes. It is produced by carbon fixation through photosynthesis, and consumed either by the storage as carbon reserves X_C, by the biosynthesis of structural biomass X_S, and by activity respiration that provides the energy needed to perform any metabolic process.

– X_C (mmol C m^{-3}) is the intracellular carbon reserve as carbohydrates. A fast exchange occurs between this reserve and X_M; X_C is thus the storage compartment for the part of fixed carbon that is not immediately used by the metabolism.

– X_N (mmol N m^{-3}) is the intracellular nitrogen reserve as glutamine, available to build up the structural biomass X_S. It represents the internal N-status. Glutamine has fixed C:N ratio.

– X_S (mmol C m^{-3}) is the structural carbon; it has fixed C:N ratio.

– X_{nit} (mol N mol C^{-1} h^{-1}) is the nitrogenase activity. This representation is similar to Stephens et al. (2003).

This description leads to a mass balance (Fig. 1) model that represents the growth dynamics of the *Trichodesmium* population. The complete model will be fully described in a forthcoming paper; here we focus on the dynamics of the N_2 fixation mechanism. N_2 fixation is decomposed as follows:

– The actual N_2 fixation rate, f (mmol N m^{-3} h^{-1}), depends on the potential activity of the nitrogenase (X_{nit}) and is proportional to the structural biomass (X_S). It further relies on the availability of the internal carbon reserves (limitation coefficient l_{X_M}), and is conversely proportional to the internal N-status (limitation coefficient l_{X_N}):

$$f = X_{nit} . X_S . l_{X_M} . (1 - l_{X_N}) \quad (1)$$

l_{X_M} (*resp.* l_{X_N}) is a function of the relative proportion of the small metabolites (*resp.* nitrogen reserves) compared to the minimum cellular C (*resp.* N) quota, following a sigmoid relationship. With the formulation (1), changes in N_2 fixation occur in relation to the photosynthetic activity of cells, via the dynamics of the carbon reserves.

– Nitrogenase activity (X_{nit}) is synthesised and decays. As the process is energetically expensive, its synthesis is promoted by the energy available for the enzyme synthesis (light and l_{X_M}), and decreases in response to (1)– the availability of nitrate in the environment and (2)– the availability of the intracellular nitrogen reserves (l_{X_N}). *Thus* N-uptake is given priority to N fixation.

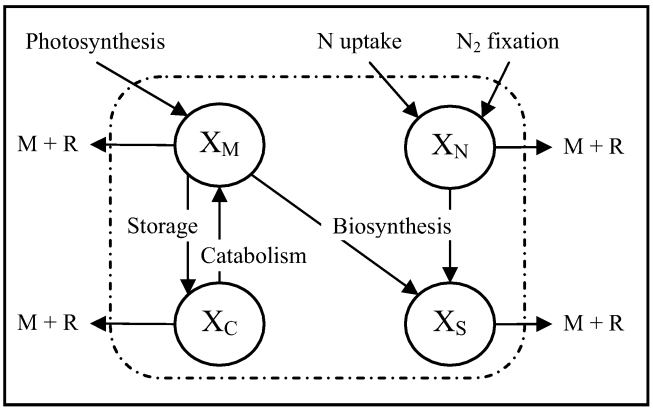


Figure 1: Schematic representation of the Mass Transfers. M = mortality and R = respiration

Thus, N_2 fixation is not directly related to photosynthesis but relies on the availability of carbon reserves, on the amount of active enzyme in cells, and is also influenced by internal N reserves. This allows N_2 fixation to go on when the sun sets, provided energy reserves are available. This formulation is therefore also applicable to other fixing strategies, as for instance the unicellular *Synechococcus* sp. that temporally separates N_2 fixation from the oxygenic photosynthesis (fixation during the night).

In what follows we discuss several dynamic simulations in qualitative terms. The simulations are run under various light intensities and different nitrogen sources and in varying experimental setups (i.e. batch culture, dilution experiments).

RESULTS AND DISCUSSION

The model reproduces the daily dynamics of the fixation process that can be observed in cultures (unpublished data), in relation to the light regime. Under Light/Dark (L/D) regimes,

nitrogenase activity increases rapidly during the first half of the light period. A maximum is observed at mid- L phase or before, followed by a slower decrease during the second half of the light phase.

Simulations of batch experiments exhibit a transition from a growth based on DIN-uptake towards the onset of N_2-fixation when nutrients are almost depleted. N_2 fixation alone can support the growth when DIN is absent. Simulations of dilution experiments reveal that N fixation does not occur if sufficient inorganic nutrients are supplied. However, when simulating an experiment with a low N income, nitrogen fixation occurs together with the uptake (Fig. 2).

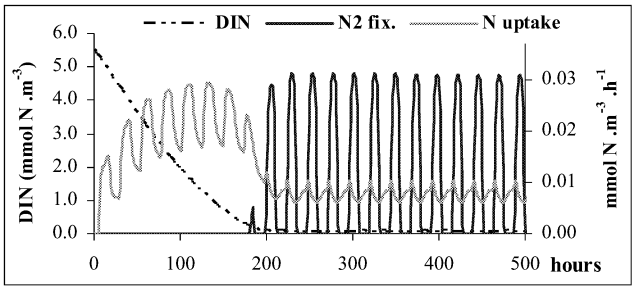


Figure 2: Simulation of N_2 fixation and N uptake in a Dilution experiment. PAR= 200 µmol phot. m^{-2} s^{-1}, D= 0.075 h^{-1}, DIN input= 1.1 mmol N m^{-3}

The steady state is reached after the onset of the nitrogen fixation and depends on the conditions of light as the irradiance drives the amount of energy available for the N_2 fixation process. Figure 3 shows that relative N reserves increase with available light, indicating a higher N_2 fixation activity.

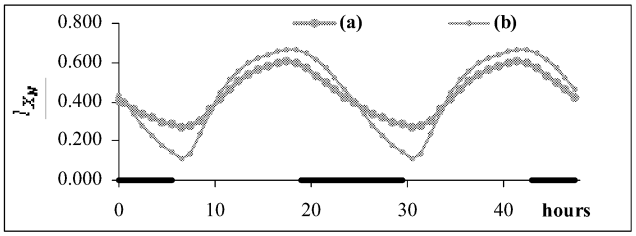


Figure 3: Steady State Limitation Coefficient l_{X_N} in 2 dilution experiments with an average irradiance of (a): 200 and (b): 300 µmol phot. m^{-2} s^{-1}. D= 0.075 h^{-1} and DIN input = 1.1 mmol N m^{-3}. Horizontal bars indicate the dark period

Growth of *Trichodesmium* is potentially independent of NO_3 and NH_4 in its environment. However, as N_2 fixation is more costly than the uptake, the biomass that can be produced, under a given irradiance, will vary depending on the mode of N-nutrition. The cyanobacterium can thus sustain a higher growth rate when inorganic nutrients are available. Finally, the steady state also depends on the concentration of the N input (Fig. 4).

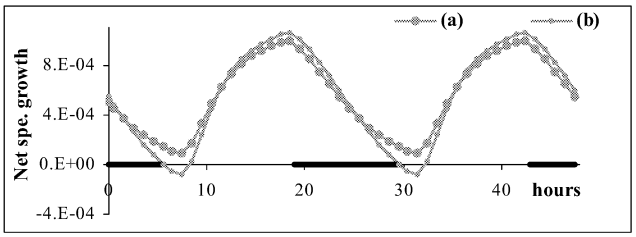


Figure 4: Steady state Net Specific Growth rate (h^{-1}) in 2 Dilution experiments with a DIN input of (a): 1 and (b): 10 mmol N m^{-3}. D= 0.075 h^{-1}; light = 300 µmol phot. m^{-2} s^{-1}. Horizontal bars indicate the dark period

CONCLUSION

The dynamics of the N_2-fixation process is complex as it involves the influence of external but also internal factors. The combined representation of the internal reserves of carbon and nitrogen and the description of the potential activity of the nitrogenase are essential parts of the model formulation. The mathematical model may provide a scientific reflection tool conceptualising the ecological determinants of the nitrogen fixation process.

REFERENCES

Bergman, B. and Carpenter, E.J, (1991) J. Phycol. **27**(2): 158-165.

Bergman, B., Gallon, J.R., Rai, A.N. and Stal, L.J. (1997) FEMS Microbiol. Rev., **19**(3): 139-185.

Berman-Franck, I., Cullen, J.T., Shaked, Y., Sherrell, R.M. and Falkowski, P.G. (2001) Limnol. Oceanogr. **46**(6): 1249-1260.

Capone, D.G., Zehr, J.P., Paerl, H.W., Bergman, B. and Carpenter, E.J. (1997) Science, **276**: 1221-1229.

Droop, M.R. (1973) Phycol., **9**: 264-272.

Fennel, K., Spitz, Y.H., Letelier, R.M., Abbott, M.R. and Karl, D.M. (2002) Deep-Sea Res. PT II **49**: 149-174.

Flynn, K.J. and Fasham, M. J. R. (2003) J. Plankton Res. **25**: 83-92.

Gallon, J.R. (1992) New Phytol. **122**: 571-609.

Gallon, J.R. (2001) Plant and Soil **230**: 39-48.

Geider, R.J., MacIntyre, H.L. and Kana, T.M.(1998) Limnol. Oceanogr. **43**(4): 679-694

Hood, R.R., Bates, N.R., Capone, D.G. and Olson, D.B. (2001) Deep-Sea Res. PT II **48**: 1609-1648.

Karl, D., Michaels, A., Bergman, B., Capone, D. et al (2002) Biogeochemistry **57/58**: 47-98.

Lancelot, C. and Billen, G. (1985) Adv. Aquat. Microbiol. **3**: 263-321

Lancelot, C., Veth, C., Becquevort, S. and Mathot, S. (1991) Mar. Chem. **35**: 305-324.

Meersche, Van den, K., Middelburg J.J., Soetaert, K. et al. 2004 Limnol. Oceanogr. **49**(3): 862-878.

Moore, J.K., Doney, S.C., Kleypas, J.A., Glover, D.M. and Fung, I.Y. (2002) Deep-Sea Res. PT II **49**: 403-462.

Mulholland, M.R., Ohki, K. and Capone, D.G. (2001) J. Phycol., **37**(6): 1001-1009.

Soetaert, K. Herman P.M.J, Middelburg J.J., et al. (2001) Deep-Sea Res. PT II **48**: 3141-3177.

Soetaert, K., deClippele, V. and Herman, P. (2002) Ecol. Modelling **151**(2-3): 177-193.

Stephens, N., Flynn, K.J. and Gallon, J.R. (2003) New Phytol. **160**(3): 545-555.

Tett, P. (1998) Napier Univ.

TOWARDS A FRAMEWORK FOR MULTI-SCALE MODELS OF FRESHWATER FISHES

Nicolas DUMOULIN
Laboratoire d'Ingénierie pour les Systèmes Complexes (LISC)
CEMAGREF, 24 avenue des Landais, BP 50085
63172 Aubière Cedex, France
Nicolas.Dumoulin@clermont.cemagref.fr

INTRODUCTION

Individual-based models are becoming an important tool in ecosystems modelling. The main disadvantage encountered with this kind of model is the difficulty to understand how the global dynamics emerges from individual behaviours. We first present a sample of the various processes and sub-models that could intervene in a model of freshwater fishes. Then, we introduce the need for understanding the dynamics of an individual-based model. Finally, we present a framework to help the design and the software implementation of an individual-based model of freshwater fishes.

FRESHWATER FISHES MODELING

Spatial representation

In order to model fish movements, we need a spatially explicit model, which aims to link the habitat properties with individuals. Representing the environment implies three levels of abstraction (Bian, 2003), namely the conceptual model, the spatial data model, and the spatial data structure. Here, we focus on the spatial data model, that formalizes the representation of objects (individuals) and fields (habitat properties). This representation is obtained by discretizing the landscape, in order to partition the continuous environment into a finite number of discrete parts. This discretization can be achieved in two principal ways, namely the grid model and the patch model. The grid model is the simplest because of its geometric regularity, whereas the patch model induces more computation to find the distance between two patches, and more memory to store vertex coordinates and neighbours.
The grid model is usually composed of squares, and eventually of hexagons. The patch model can be defined as a set of polygons with variable dimensions. The main advantage of the patch model is its spatial scalability, i.e. the granularity may change according to the requested resolution on a local region.

Several spatial data models could be used for a spatially-explicit model of freshwater fishes, each of them with his own strength and weakness. However, this model often depends on the spatial representation of the hydrological model, if any.

Freshwater fish movements

We can classify freshwater fish movements in three categories (Giske et al., 1998, Lucas and Baras, 2001) namely spawning, feeding, and seeking shelter. The modelling of these behaviours requires several different informations (biotic and abiotic) about the habitat.
In addition, fishes may move at two time scales. On the one hand, they accomplish daily moves for their feeding and fitness. These daily moves are restricted to an area called the home range, that is a characteristic particular to each species. On the other hand, they migrate in order to move from an habitat patch to another one. Indeed, fish populations are spatially subdivided in metapopulations, that are connected by migrations (Hanski, 1994). These migrations are essential for the specie subsisting. Without them, a local metapopulation risks the extinction (Keith et al., 1992, Thomas, 1994, Hein et al., 2004). Thus, the connectivity between habitat patches must be preserved. This representation of the habitat fragmentation permit to bring up a spatial data model, that consists in a network of idealized habitat patches (Hanski, 1998).

Existing models of fish movements

Many models of fish movements are using a model of habitat for conditioning behaviour of fishes. The Instream Flow Incremental Methodology/Physical HAbitat Simulation (IFIM/PHABSIM) (Milhous et al., 1989, Bovee et al., 1998) and EVHA (Ginot, 1998) are methods that enable to obtain habitat properties from instream flow quantity. As an output of these models, we could rate the habitat value from its physical properties. The main disadvantage of these methods is the restricted length of the tran-

sect (portion of a river) studied.
PHABSIM has been used for developing an individual-based model of trout (Winkle et al., 1998, Railsback et al., 1999, Railsback and Harvey, 2002). In this model, at each timestep (each day), each fish tries to maximize its fitness by choosing his destination. This fitness optimization is the core of the dynamics.

HOW TO DEAL WITH PATTERNS EMERGENCE

Individual-based modelling

An individual-based model (IBM) is a bottom-up approach which allows to represent explicitly the features of each individual and of the system, and then to observe how properties of the population emerge from the interactions at the individual level (Grimm, 1999). Compared to a state variable model, this method provides more flexibility and more convenience for integrating the descriptions of an ecosystem, and for analyzing. But, as a consequence, IBMs dynamics are more complex to understand, and simulations require more computation time, especially if the model includes stochasticity because of the necessity of replicating simulation.

Pattern-oriented modelling

A pattern is a noteworthy characteristic that can be observed in nature itself or through data extracted from nature (Grimm et al., 1996). A pattern is free from random effects and thus indicates an underlying process that generates this pattern (Levin, 1992). Pattern-oriented modelling consists in keeping as a guideline the pattern that we would like to. This is a good approach to obtain a final model reduced to the minimum.

Aggregation

In order to observe patterns, data returns from an IBM simulation should be aggregated. Aggregation of individual data enables to examine several points of view, as illustrated on figure 1, from spatial distribution of fishes in a river, we could study the behaviours of either the individuals, or cohorts, or metapopulations, or the population as a whole. This aggregation is typically obtained by data analysis, but it could also be obtained by aggregating individuals during the simulation (Scheffer et al., 1995).

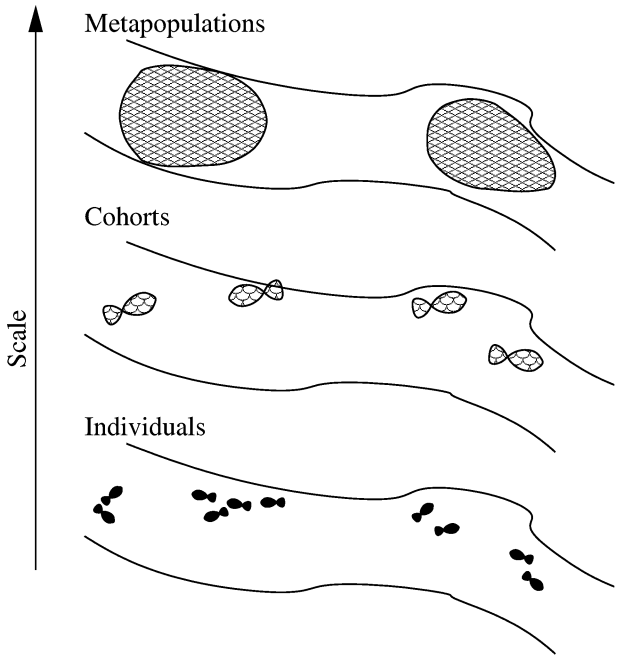


Figure 1: Several points of view on the same system at different scales..

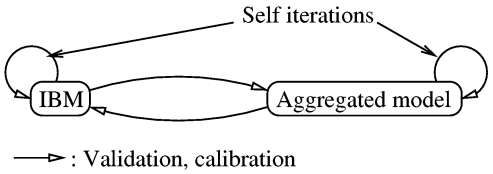


Figure 2: Additional iterations between the IBM and the aggregate model..

UNIFIED FRAMEWORK

According to the heterogeneity of data, data models, and the diversity of sub-models that could be included (population dynamics, predation, hydrological model, ...), the purpose of our work is to offer a technical solution to go towards a compatibility between the different sub-models. For that, we propose a library of classes dedicated to modelling of freshwater fishes, that will offer a common spatial model as abstract classes.
In the future, the aim is to allow for an IBM and an aggregate model to interoperate, in order to benefit from the advantages of each one. Indeed, both models give another way of studying the dynamics of a system, that could help for validating and calibrating. A modeler needs several iterations to build his model (see figure 2), but with a second model, he has got an additional way to do this.

SUMMARY AND CONCLUSION

A lot of models of freshwater fishes have been developed, and it would be interesting if some of these could

be coupled (for example a model of population dynamics and a model of movement). However, coupling models is a hard task, our aim is also to make this step easier.

REFERENCES

Bian, L. 2003. "The Representation of the Environment in the Context of Individual-Based Modeling.". *Ecological Modelling*, 159(2-3):279–296.

Bovee, K. D.; Lamb, B. L.; Bartholow, J. M.; Stalnaker, C. B.; Taylor, J.; and Henriksen, J. 1998. "Stream Habitat Analysis Using the Instream Flow Incremental Methodology.".Technical report, Fort Collins, CO.

Ginot, V. 1998. "EVHA 2.0 Evaluation de L'habitat Physique Des Poissons En Rivière; Guide de L'utilisateur.".Technical report, Cemagref, Ministère de l'intérieur.

Giske, J.; Huse, G.; and Fiksen, Ø. 1998. "Modelling Spatial Dynamics of Fish.". *Reviews in Fish Biology and Fisheries*, 8(1):57–91.

Grimm, V. 1999. "Ten Years of Individual-Based Modelling in Ecology: What Have We Learned and What Could We Learn in the Future?". *Ecological Modelling*, 115(2-3):129–148.

Grimm, V.; Frank, K.; Jeltsch, F.; Brandl, R.; Uchmanski, J.; and Wissel, C. 1996. "Pattern-Oriented Modelling in Population Ecology.". *The Science of The Total Environment*, 183(1-2):151–166.

Hanski, I. 1994. "A Practical Model of Metapopulation Dynamics.". *Journal of Animal Ecology*, 63(1):151–162.

Hanski, I. 1998. "Metapopulation Dynamics.". *Nature*, 396(6706):41–49.

Hein, S.; Pfenning, B.; Hovestadt, T.; and Poethke, H. J. 2004. "Patch Density, Movement Pattern, and Realised Dispersal Distances in a Patch-Matrix Landscape–a Simulation Study.". *Ecological Modelling*, 174(4):411–420.

Keith, P.; Allardi, J.; and Moutou, B. 1992. "Livre Rouge Des Espèces Menacées de Poissons D'eau Douce de France Et Bilan Des Introductions.".Paris.

Levin, S. A. 1992. "The Problem of Pattern and Scale in Ecology.". *Ecology*, 73(6):1943–1967.

Lucas, M. C. and Baras, E. 2001.*Migration of Freshwater Fishes.*Blackwell Science Ltd, Oxford.

Milhous, R. T.; Updike, M. A.; and Schneider, D. M. 1989.*Physical Habitat Simulation System Reference Manual – Version II.*U.S. Fish and Wildlife Service, Washington, DC, biological report 89(16) edition.

Railsback, S. F. and Harvey, B. C. 2002. "Analysis of Habitat-Selection Rules Using an Individual-Based Model.". *Ecology*, 83(7):1817–1830.

Railsback, S. F.; Lamberson, R. H.; Harvey, B. C.; and Duffy, W. E. 1999. "Movement Rules For Individual-Based Models of Stream Fish.". *Ecological Modelling*, 123(2-3):73–89.

Scheffer, M.; Baveco, J. M.; DeAngelis, D. L.; Rose, K. A.; and Nes, E. H. V. 1995. "Super-Individuals a Simple Solution For Modelling Large Populations On an Individual Basis.". *Ecological Modelling*, 80(2-3):161–170.

Thomas, C. D. 1994. "Extinction, Colonization, and Metapopulations: Environmental Tracking By Rare Species.". *Conservation Biology*, 8(2):373–378.

Winkle, W. V.; Jager, H. I.; Railsback, S. F.; Holcomb, B. D.; Studley, T. K.; and Baldrige, J. E. 1998. "Individual-Based Model of Sympatric Populations of Brown and Rainbow Trout For Instream Flow Assessment: Model Description and Calibration.". *Ecological Modelling*, 110(2):175–207.

NICOLAS DUMOULIN got his MSc (french DEA) in Computer Science in 2003 from Rennes University in 2003. He's now pursuing his Ph.D. studies in the Laboratory of Engineering for Complex Systems at Cemagref. His research activities deal with the software implementation of an architecture to assist the developpement and analyze of individual-based models of freshwater-fishes.

ANALYTICAL AND NUMERICAL MODELLING TECHNIQUES

STIFFNESS MODELLING OF RUBBER ENGINE MOUNTS BY POLYNOMIAL NEURAL NETWORKS AND GENETIC ALGORITHMS

N. Nariman-zadeh
Department of Mechanical Engineering,
Engineering Faculty, The University of Guilan
P.O. Box 3756, Rasht, IRAN
E-mail: nnzadeh@guilan.ac.ir

J. Marzbanrad, A. Jamali
Department of Automotive Engineering,
Iran University of Science & Technology
Tehran, Iran

KEYWORDS
Engine mount, GMDH, Genetic Algorithms, Finite Element Method (FEM).

ABSTRACT

Group Method of Data Handling (GMDH) is used to obtain a polynomial model among stiffness and geometric parameters of an engine mount. In this approach, Genetic Algorithms (GAs) are employed for optimal design of connectivity configuration of the GMDH-type neural networks. The aim of such modelling is to show how the stiffness of an engine mount varies with the variation of geometric parameters. In this way, the stiffness training data are obtained for 36 different bush type engine mounts by using the Finite Element Analysis (FEA).

INTRODUCTION

The engine is the largest concentrated mass in a vehicle which will cause vibration if it is not properly isolated and constrained. An ideal engine mount system isolates engine vibration caused by engine unbalanced forces and prevents engine bounce from shock excitation. The modern engine mounting systems have been successfully used to isolate the driver and passenger from both noises and vibrations generated from the engine (Yu et al. 2001). The proper design of rubber mounts may be the most effective engineering approach to improve the ride. The analysis of engine mounting rubber components should be accompanied by the vibration analysis of the engine mount system. It is necessary not only to know what rubber properties are and where to place the mounts, but also to determine the optimum design of a rubber part to achieve the desired properties along with the required load-bearing capacity resulting from the system vibration analysis (Bernuchon 1984).

Recently, computer-aided nonlinear analyses by finite element methods have been applied in the actual design of rubbers and plastics used as engine mounts (Kim et al. 1993). Rubbers are used in many parts of the automobile, but most of the rubber components have been designed by experiences or experiments because of the complicated characteristics of rubber materials. Considering the waste of production time and material losses resulting from these approaches, more systematic and analytical approaches are desirable.

The design procedure of an engine mount is shown in figure (1). From the vibration analysis of the engine mount system, the primary stiffness values in three principal directions and the positions of mounts are determined so that the amount of vibrational energy transmitted to the driver is minimized (Ford 1985). The secondary stiffness and gap size are determined from the static nonlinear gap analysis for the severe load cases. The shape optimization in the next step of design procedure and can be performed if a mathematical model among stiffness and geometric parameters are known for a certain rubber bush type engine mount. In this stage the shape of engine mount is designed according to the characteristics determined from previous steps of design procedure.

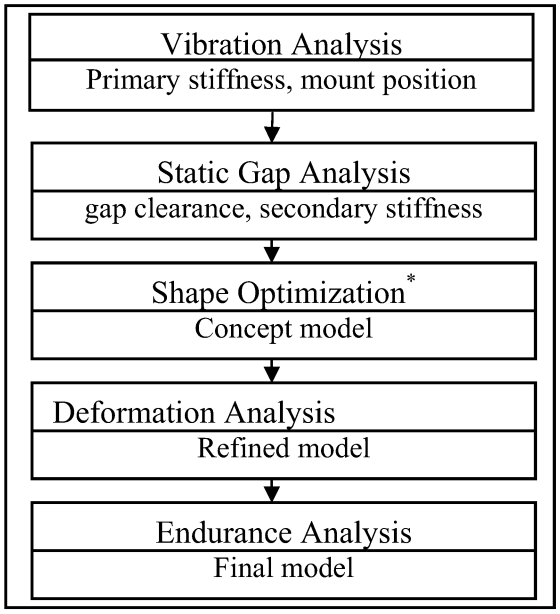


Figure 1: Design procedure of an engine mount

In this work, an optimum mathematical equation is found as a model between stiffness and geometric parameters of engine mount. Finally, the design procedure is completed and tested by next two stages for deformation and endurance analysis. Therefore, it can be seen that the

optimum design procedure requires such model identification of rubber engine mount.

System identification techniques are applied in many fields in order to model and predict the behaviors of unknown and/or very complex systems based on given input-output data (Astrom and Eykhoff 1971). Theoretically, in order to model a system, it is required to understand the explicit mathematical input-output relationship precisely. Such explicit mathematical modelling is, however, very difficult and is not readily tractable in poorly understood systems. Alternatively, soft-computing methods (, which concern computation in imprecise environment, have gained significant attention.

The main components of soft computing, namely, fuzzy-logic, neural network, and genetic algorithm have shown great ability in solving complex non-linear system identification and control problems. Several research efforts have been expended to use evolutionary methods as effective tools for system identification (Sanchez et al. 1997; Iba et al 1996). Among these methodologies, Group Method of Data Handling (GMDH) algorithm is self-organizing approach by which gradually complicated models are generated based on the evaluation of their performances on a set of multi-input-single-output data pairs (i=1, 2, …, M). The GMDH was firstly developed by Ivakhnenko (Ivakhnenko 1971) as a multivariate analysis method for complex systems modelling and identification. In this way, GMDH was used to circumvent the difficulty of knowing a priori knowledge of mathematical model of the process being considered. In other words, GMDH can be used to model complex systems without having specific knowledge of the systems. The main idea of GMDH is to build an analytical function in a feedforward network based on a quadratic node transfer function (Farlow 1984) whose coefficients are obtained using regression technique. In fact, real GMDH algorithm in which model coefficients are estimated by means of the least squares method has been classified into complete induction and incomplete induction, which represent the combinatorial (COMBI) and multilayered iterative algorithms (MIA), respectively (Farlow 1984). In recent years, however, the use of such self-organizing network leads to successful application of the GMDH-type algorithm in a broad range area in engineering, science and economics (Farlow 1984; Nariman-zadeh et al. 2003).

In this paper, stiffness variation with geometric parameters of a typical engine mount is modelled using GMDH-type neural networks. In this way, 36 different geometry bush type engine mounts are first analyzed by FEA to obtain their corresponding stiffness in 2 orthogonal directions (k_x, k_y). The parameters involved in the geometry of an engine mount are regarded as inputs whilst the computed stiffness values by FEA are regarded as outputs in such modelling. The input-output relationship in the form of polynomial models of the engine mount will be then found by GMDH-type neural networks. Such neural network identification process needs, in turn, some optimization methods to find the best network architecture. In this way, genetic algorithms are deployed in a new approach to design the whole architecture of the GMDH-type neural networks, i.e., the number of neurons in each hidden layer and their connectivity configuration, in combination with using Singular Value Decomposition (SVD) to find the optimal set of appropriate coefficient of quadratic expressions for modelling of engine mount stiffness. The connectivity configuration is not limited to the adjacent layers, as have already proposed in (Nariman-zadeh et al. 2003), so that general structure GMDH-type neural networks, in which neurons in any layers can be connected to each other, are evolved. Furthermore, a new yet simple encoding scheme has been introduced to present the genotype of general structure GMDH-type neural networks so that the length of chromosome is kept minimal and thus leads to simpler evolutionary operations including crossover and mutation. Such encoding scheme must allow for producing and/or switching both generalized and simplified networks using the genetic operators during evolutionary process.

MODELLING USING GMDH NEURAL NETWORKS

By means of GMDH algorithm a model can be represented as set of neurons in which different pairs of them in each layer are connected through a quadratic polynomial and thus produce new neurons in the next layer. Such representation can be used in modelling to map inputs to outputs. The formal definition of the identification problem is to find a function $\hat{f}$ so that can be approximately used instead of actual one, f, in order to predict output $\hat{y}$ for a given input vector $X = (x_1, x_2, x_3, ..., x_n)$ as close as possible to its actual output y. Therefore, given M observation of multi-input-single-output data pairs so that

$$y_i = f(x_{i1}, x_{i2}, x_{i3}, ..., x_{in}) \quad (i=1,2,...,M)$$

it is now possible to train a GMDH-type neural network to predict the output values $\hat{y}_i$ for any given input vector $X = (x_{i1}, x_{i2}, x_{i3}, ..., x_{in})$, that is

$$\hat{y}_i = \hat{f}(x_{i1}, x_{i2}, x_{i3}, ..., x_{in}) \qquad (i=1,2,...,M).$$

The problem is now to determine a GMDH-type neural network so that the square of difference between the actual output and the predicted one is minimised, that is

$$\sum_{i=1}^{M} [\hat{f}(x_{i1}, x_{i2}, x_{i3}, ..., x_{in}) - y_i]^2 \rightarrow \min .$$

General connection between inputs and output variables can be expressed by a complicated discrete form of the Volterra functional series in the form of

$$y = a_0 + \sum_{i=1}^{n} a_i x_i + \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j + \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} a_{ijk} x_i x_j x_k + ... \quad (1)$$

which is known as the Kolmogorov-Gabor polynomial (Sanchez et al. 1997; Iba et al. 1996; Ivakhnenko 1971; Farlow 1984; Nariman-zadeh et al. 2003). This full form of mathematical description can be represented by a system of partial quadratic polynomials consisting of only two variables (neurons) in the form of

$$\hat{y} = G(x_i, x_j) = a_0 + a_1 x_i + a_2 x_j + a_3 x_i x_j + a_4 x_i^2 + a_5 x_j^2 \quad (2)$$

In this way, such partial quadratic description is recursively used in a network of connected neurons to build the general mathematical relation of inputs and output variables given in equation (1). The coefficient a_i in equation (2) are calculated using regression techniques (Farlow 1984; Nariman-zadeh et al. 2003) so that the difference between actual output, *y,* and the calculated one, $\hat{y}$, for each pair of x_i, x_j as input variables is minimized. Indeed, it can be seen that a tree of polynomials is constructed using the quadratic form given in equation (2) whose coefficients are obtained in a least-squares sense. In this way, the coefficients of each quadratic function G_i are obtained to optimally fit the output in the whole set of input-output data pair, that is

$$E = \frac{\sum_{i=1}^{M} (y_i - G_i())^2}{M} \rightarrow \min \quad . \quad (3)$$

In the basic form of the GMDH algorithm, all the possibilities of two independent variables out of total n input variables are taken in order to construct the regression polynomial in the form of equation (2) that best fits the dependent observations (y_i, *i=1, 2, ..., M*) in a least-squares sense. Consequently, $\binom{n}{2} = \frac{n(n-1)}{2}$ neurons will be built up in the first hidden layer of the feedforward network from the observations $\{(y_i, x_{ip}, x_{iq})$; (*i=1, 2, ..., M*)$\}$ for different $p, q \in \{1,2,...,n\}$. In other words, it is now possible to construct M data triples $\{(y_i, x_{ip}, x_{iq})$; (*i=1, 2, ..., M*)$\}$ from observation using such $p, q \in \{1,2,...,n\}$ in the form

$$\left[\begin{array}{cc|c} x_{1p} & x_{1q} & y_1 \\ x_{2p} & x_{2q} & y_2 \\ \hline x_{Mp} & x_{Mq} & y_M \end{array}\right].$$

Using the quadratic sub-expression in the form of equation (2) for each row of M data triples, the following matrix equation can be readily obtained as

$$A\,\mathbf{a} = Y$$

where $\mathbf{a}$ is the vector of unknown coefficients of the quadratic polynomial in equation (2)

$$\mathbf{a} = \{a_0, a_1, a_2, a_3, a_4, a_5\} \quad (4)$$

and

$$Y = \{y_1, y_2, y_3, ..., y_M\}^T$$

is the vector of output's value from observation. It can be readily seen that

$$A = \left[\begin{array}{cccccc} 1 & x_{1p} & x_{1q} & x_{1p}x_{1q} & x_{1p}^2 & x_{1q}^2 \\ 1 & x_{2p} & x_{2q} & x_{2p}x_{2q} & x_{2p}^2 & x_{2q}^2 \\ \hline 1 & x_{Mp} & x_{Mq} & x_{Mp}x_{Mq} & x_{Mp}^2 & x_{Mq}^2 \end{array}\right]$$

The least-squares technique from multiple-regression analysis leads to the solution of the normal equations in the form of

$$\mathbf{a} = (A^T A)^{-1} A^T Y \quad (5)$$

which determines the vector of the best coefficients of the quadratic equation (2) for the whole set of M data triples. It should be noted that this procedure is repeated for each neuron of the next hidden layer according to the connectivity topology of the network. However, such a solution directly from normal equations is rather susceptible to round off errors and, more importantly, to the singularity of these equations.

APPLICATION OF SVD TO THE DESIGN OF GMDH-TYPE NETWORKS

Singular Value Decomposition (SVD) is the method for solving most linear least square problems that some singularities may exist in the normal equations. The SVD of a matrix, $A \in \Re^{M \times 6}$, is a factorisation of the matrix into the product of three matrices, column-orthogonal matrix $U \in \Re^{M \times 6}$, diagonal matrix $W \in \Re^{6 \times 6}$ with non-negative elements (singular values), and orthogonal matrix $V \in \Re^{6 \times 6}$ such that

$$A = U\,W\,V^T$$

The problem of optimal selection of vector of the coefficients in equations (4) (5) is firstly reduced to finding the modified inversion of diagonal matrix *W* (in which the reciprocals of zero or near zero singulars (according to a threshold) are set to zero. Then, such optimal $\mathbf{a}$ is calculated using the following relation

$$\mathbf{a} = V\,[diag(1/w_j)]\,U^T\,Y$$

Such parametric identification problem is part of the general problem of modelling when structure identification is considered together with the parametric identification problem simultaneously. In this work, a new encoding scheme is presented in an evolutionary approach for simultaneous determination of structure and parametric identification of GMDH neural networks.

APPLICATION OF GA IN THE TOPOLOGY DESIGN OF GMDH-TYPE NEURAL NETWORKS

Evolutionary methods such as genetic algorithms have been widely used in different aspects of design in neural networks because of their unique capabilities of finding a global optimum in highly multi-modal and/or non-differentiable search space (Iba et al. 1996; Yao 1999). Such stochastic methods are commonly used in the training of neural networks in terms of associated weights or coefficients which have successfully performed better than

traditional gradient-based techniques (Nariman-zadeh et al. 2003). The literature shows that a wide range of evolutionary design approaches either for architectures or for connection weights separately, in addition to efforts for them simultaneously (Yao 1999) In the most GMDH-type neural network, neurons in each layer are only connected to neuron in its adjacent layer as it was the case in Methods I and II previously reported in (Nariman-zadeh 2003). Taking this advantage, it was possible to present a simple encoding scheme for the genotype of each individual in the population as already proposed by authors (Nariman-zadeh 2003).

However, in order to make it more general, it is necessary to remove such restriction. In such representation, neurons in different layers including the input layer can be connected to others far away, not only in the very adjacent layers. In addition, the genotype of these networks must be encoded in such a way to include the restricted networks, namely, conventional structure GMDH-type neural networks (Nariman-zadeh et al. 2003). The encoding schemes in generalized GMDH neural networks must demonstrate the ability of representing different length and size of such neural networks. Moreover, the ability of changing building blocks of information using crossover and mutation operators must be preserved in such coding representation. In the next sections, the encoding scheme of the newly developed generalized GMDH neural networks is discussed.

The Genome Representation of Generalized GMDH Neural Networks

In the generalized GMDH neural networks, neurons connections can occur between different layers which are not necessarily very adjacent ones, unlike the conventional structure GMDH neural networks in which such connections only occur between adjacent layers. For example, a network structure which depicted in figure (2) shows such connection of neuron ***ad*** directly to the output layer. Consequently, this generalisation of network's structure can evidently extend the performance of generalized GMDH neural networks in modelling of real-world complex processes. Such generalization is accomplished by repeating the name of the neuron which directly passing the next layers. In figure (2), neuron ***ad*** in the first hidden layer is connected to the output layer by directly going through the second hidden layer. Therefore, it is now very easy to notice that the name of output neuron (network's output) includes ***ad*** twice as ***abbcadad***. In other words, a virtual neuron named ***adad*** has been constructed in the second hidden layer and used with ***abbc*** in the same layer to make the output neuron ***abbcadad*** as shown in the figure (2). It should be noted that such repetition occurs whenever a neuron passes some adjacent hidden layers and connects to another neuron in the next 2^{nd}, or 3^{rd},or 4^{th},or … following hidden layer. In this encoding scheme, the number of repetition of that neuron depends on the number of passed hidden layers, ñ, and is calculated as $2^{\tilde{n}}$. It is easy to realize that a chromosome such as ***abab bcbc***, unlike chromosome ***abab acbc*** for example, is not a valid one in GS-GMDH networks and has to be simply re-written as ***abbc***.

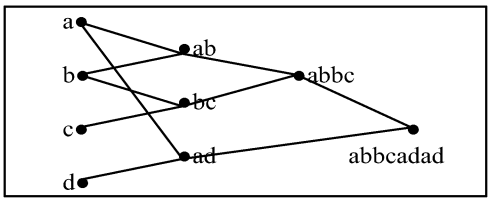


Figure 2: A Generalized GMDH Network Structure of a Chromosome

Genetic Operators for Generalized GMDH Network Reproduction

The genetic operators of crossover and mutation can now be implemented to produce two offsprings from two parents. The natural roulette wheel selection method is used for choosing two parents producing two offsprings. The crossover operator for two selected individuals is simply accomplished by exchanging the tails of two chromosomes from a randomly chosen point as shown in figure (3). It should be noted, however, such a point could only be chosen randomly from the set $\{2^1, 2^2, ..., 2^{n_l+1}\}$ where n_l is the number of hidden layers of the chromosome with the smaller length. It is very evident from figures (3) and (4) that the crossover operation can certainly exchange the building blocks information of such generalized GMDH neural networks so that the two types of generalized GMDH and conventional GMDH-type neural networks can be converted to each other, as can be seen from figure (4). In addition, such crossover operation can also produce different length of chromosomes which in turn leads to different size of either generalized GMDH or conventional GMDH network structures. Similarly, the mutation operation can contribute effectively to the diversity of the population. This operation is simply accomplished by changing one or more symbolic digits as genes in a chromosome to other possible symbols, for example, *abbcadad* to *abbccdad*. It is very evident that mutation operation can also convert a generalized GMDH network to a conventional GMDH network or vice versa. It should be noted that such evolutionary operations are acceptable provided a valid chromosome is produced. Otherwise, these operations are simply repeated until a valid chromosome is constructed.

The incorporation of genetic algorithm into the design of such GMDH-type neural networks starts by representing each network as a string of concatenated sub-strings of alphabetical digits. The fitness, (Φ), of each entire string of symbolic digits which represents a GMDH-type neural network to model explosive cutting process is evaluated in the form

$$\Phi = 1/E$$

where E, is the mean square of error given by equation (3), is minimized through the evolutionary process by maximizing the fitness Φ. The evolutionary process starts

by randomly generating an initial population of symbolic strings each as a candidate solution. Then, using the aforementioned genetic operations of roulette wheel selection, crossover and mutation, the entire populations of symbolic strings to improve gradually. In this way, GMDH-type neural network models of engine mount stiffnesses with progressively increasing fitness, Φ, are produced until no further significant improvement is achievable.

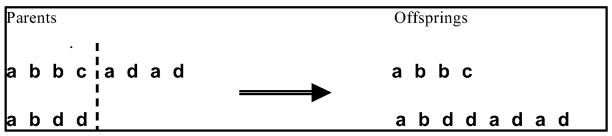


Figure 3: Crossover operation for two individuals in generalized GMDH networks

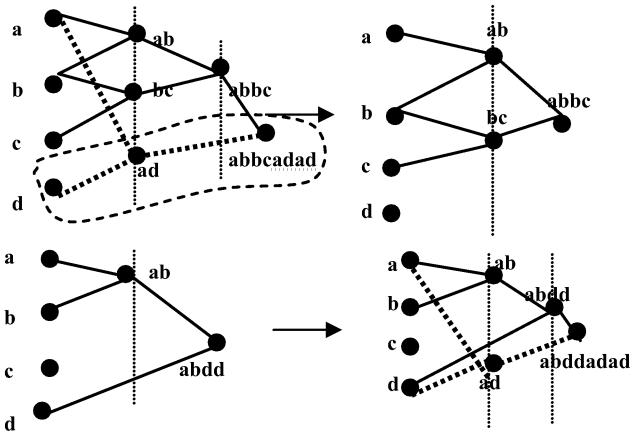


Figure 4: Crossover Operation on two Generalized GMDH Networks

MODELLIMG AND ANALYSIS OF ENGINE MOUNT

A bush type engine mount is considered in this study. Figure (5) shows the shape of the engine mount of a passenger car and its geometry. For convenience, the axis of local coordinates shown in figure (5) will be used throughout this paper. There are six geometric parameters to define the shape of bush type engine mounts, as shown in figure (5b). However, r_i, r_o and θ are known from the layout design and are fixed for a certain engine mount. Therefore, three parameters, t_r, t_s and t_z, are considered as design variables.

The stiffness values which are derived from the system vibration analysis are known as dynamic stiffnesses. However, the static stiffness can be simply computed by

$$k_d = \eta\, k_s$$

where k_d and k_s are the dynamic and static stiffness, respectively, and η is a correction factor (Kim et al. 1992). The correction factor is generally in the range from 1.2 to 1.6. In this paper, the static stiffness is modeled and dynamic stiffness can be consequently computed by a proper value of correction factor which can be selected for a particular rubber material.

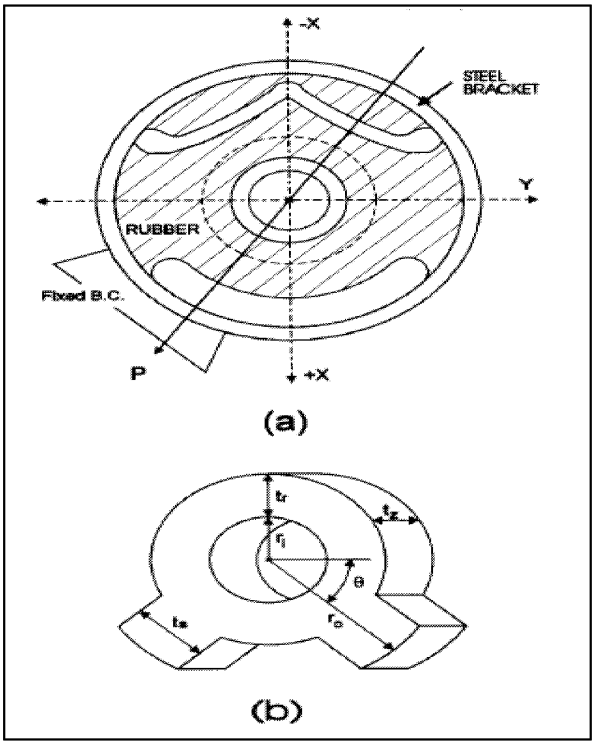


Figure 5: A schematic diagram of a bush engine mount (a) General shape (b) Characterized shape for modelling

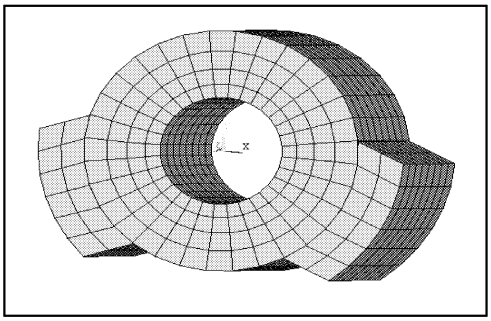


Figure 6: Finite element model of engine mount

Since the strain is relatively small in this analysis, the classical Mooney-Rivlin form of the strain energy is sufficient to describe the fully incompressible hyper elastic material behavior (Rivlin 1992).The Mooney-Rivlin form of strain energy, U, is expressed by

$$U = C_{10}(I_1 - 3) + C_{01}(I_2 - 3)$$

Where I_1 and I_2 are the first and second strain invariants, respectively. The coefficients C_{10} and C_{01} are determined from the results of the uniaxial tension test. For a special engine mount used in this paper, these values are $C_{10} = 0.03622$ and $C_{01} = -0.00335$. These values are then used for the FEA to obtain the stiffness in different directions. The range of design variables which have been used in this study are

$$10 \le t_r \le 15, \quad 25 \le t_s \le 40, \quad 25 \le t_z \le 40$$

Finite element model (FEM) of engine mount with different dimensions, in the ranges of design variables, are built. A typical FEM of engine mount is shown in figure (6). In these analyses, the fixed geometry values are $r_i = 10\,\text{mm}$, $r_o = 30\,\text{mm}$, $\theta = 12\,\text{degree}$, and the Mooney-Rivlin coefficients expected from stress-strain relationship have been given as $C_{10} = 0.03622$ and $C_{01} = -0.00335$.

THE ENGINE MOUNT GMDH NEURAL NETWORK MODELING

In this paper, 36 different geometry engine mount have been considered in the FEA in order to obtain their stiffness values in 3-orthogonal directions. The results of the values of stiffness together with their corresponding geometry dimensions are shown in table (1).

Table1: Inputs-outputs Results from FEM

No.	tr	ts	tz	ky	kx
1	10	25	25	39.13	33.33
2	10	25	30	46.262	39.6
3	10	25	35	59.429	49.76
4	10	25	40	68.182	58.01
5	10	30	25	41.475	35.43
6	10	30	35	59.639	49.75
7	10	30	40	66.242	59.43
8	10	35	25	50	36.73
9	10	35	30	59.281	44.8
10	10	35	40	77.778	59.26
11	12.5	25	25	36	30.2
12	12.5	25	30	41.509	36.51
13	12.5	25	35	47.895	42.92
14	12.5	32	25	38.298	30.3
15	12.5	32	30	42.718	36.07
16	12.5	32	35	52.907	43.13
17	12.5	32	40	61.047	50
18	12.5	40	30	43.137	38.1
19	12.5	40	35	49.727	43.19
20	12.5	40	40	57.252	51.22
21	15	25	25	35.019	31.69
22	15	25	30	42.581	38.52
23	15	25	40	57.252	52.24
24	15	32	25	35.857	31.03
25	15	32	30	40.741	39.6
26	15	32	35	51.131	43.7
27	15	32	40	54.688	52.14
28	15	40	30	36.82	35.12
29	15	40	35	46.501	41.39
30	15	40	40	53.846	51.4
31	10	30	30	49.254	43.04
32	10	35	35	69.333	52.79
33	12.5	25	40	61.404	53.3
34	12.5	40	25	34.749	42.25
35	15	25	35	49.761	45.12
36	15	40	25	34.615	30

The GMDH-type neural networks are now used for such input-output data given in table (1) to find the polynomial model of stiffness values in respect to their geometry variables. It can be simply verified that input and output parameters in such model are (t_r, t_s, t_z) and (k_x, k_y), respectively. In order to genetically design such GMDH-type neural network described in previous section a population of 25 individuals with a crossover probability of 0.7 and mutation probability of 0.07 has been used in 100 generation. The structures of the evolved 1-hidden layer GMDH-type neural network are shown in figure (7) corresponding to the genomes representation of ***abcc*** for k_y and ***abbc*** for k_x, in which ***a, b and c*** stand for t_r, t_s *and* t_z, respectively. The very good behaviour of such GMDH-type neural network models are also depicted in figures (8) and (9).

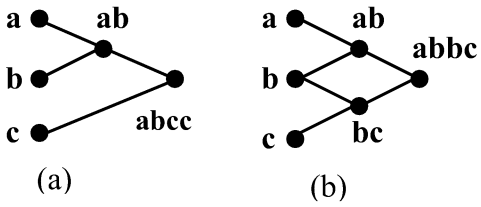


Figure 7: Evolved Structure of Generalized GMDH Neural Network: (a) K_y (b) K_x

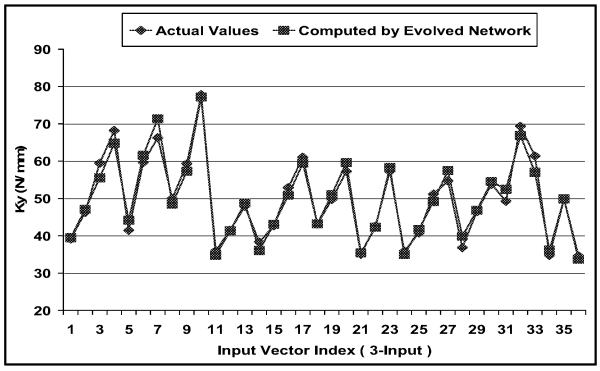


Figure 8: Comparison of Actual Stiffness at y-direction and 1-hidden Evolved GMDH Neural Model

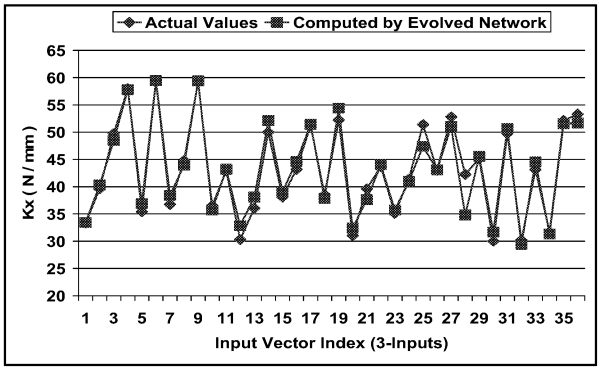


Figure 9: Comparison of Actual Stiffness at x-direction and 1-hidden Evolved GMDH Neural Model

However, in order to demonstrate the prediction ability of evolved GMDH-type neural networks, the data has been divided into two different sets, namely, training and testing sets. The training set, which consists of 30 out of 36 inputs-output data pairs, is used for training the neural network models the evolutionary method of this paper. The testing set, which consists of 6 unforeseen inputs-output data samples during the training process, is merely used for testing to show the prediction ability of such evolved GMDH-type neural network models during the training process. The structures of GMDH network are same of figure (7). The very good behaviour of such GMDH-type

neural network models are also depicted in figures (10) and (11). The corresponding polynomial representation of such model for k_y is:

$$Y_1 = 114.39 - 18.44t_r + 3.57t_s + 0.83t_r^2 - 0.01t_s^2 - 0.16t_r t_s$$

$$k_y = 109.37 - 0.84Y_1 - 3.7t_z + 0.014Y_1^2 + 0.036k_z^2 + 0.029Y_1 t_z$$

and for k_x that is:

$$Y_3 = 141.79 - 11.36t_r - 1.53t_s + 0.665t_r^2 + 0.067t_s^2 - 0.213364537297t_r t_s$$

$$Y_4 = -34.275 + 3.29t_s + 0.16t_z - 0.036t_s^2 + 0.034t_z^2 - 0.031t_s t_z$$

$$k_x = -58.06 + 1.9Y_3 + 1.057Y_4 - 0.016Y_3^2 - 0.0043Y_4^2 + 0.0067Y_3 Y_4$$

It is clearly evident that the evolved GMDH-type neural network in terms of simple polynomial equations can successfully model and predict the output of testing data which has not been used during the training process.

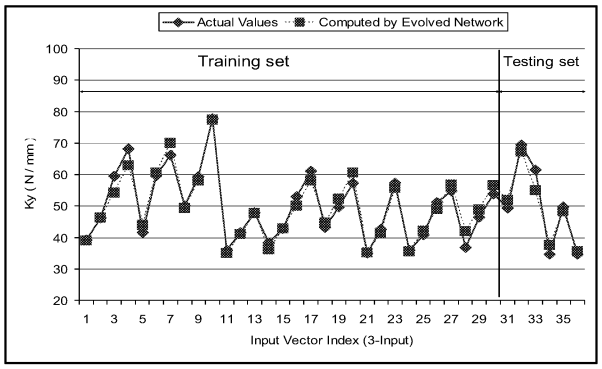


Figure 10: Comparison of Actual Stiffness at y-direction and 1-hidden Evolved GMDH Neural (Model1ing & Prediction)

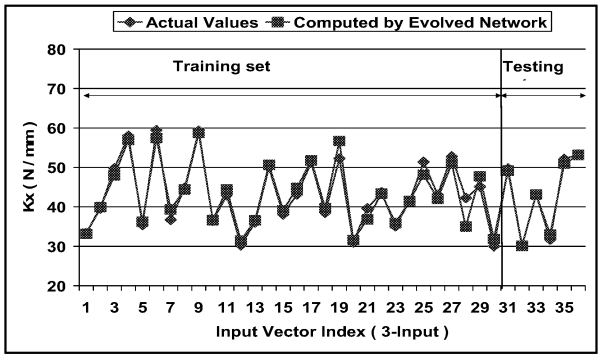


Figure 11: Comparison of Actual Stiffness at x-direction and 1-hidden Evolved GMDH Neural (Model1ing & Prediction)

CONCLUSION

Evolutionary methods for designing generalized GMDH-type networks have been proposed and successfully used for the modelling of the engine mount stiffness. In this way, it has been shown that GMDH-type networks provide effective polynomial to model and predict the stiffness of engine mount and designer can use these equations for design and select dimensions of engine mount.

Besides, a new encoding scheme has been also presented to genetically design generalized GMDH-type neural networks in which the connectivity configuration in such networks is not limited to adjacent layers, unlike the conventional GMDH-type neural networks.

References

Astrom, K.J. and P. Eykhoff. 1971. "System identification, a survey", *Automatica* 7-123-62.

Bernuchon, M. 1984. "A New Generation of Engine Mounts", *SAE Technical Paper Series*, 840259.

Farlow, S.J., ed. 1984. *"Self-organizing Method in Modelling: GMDH type algorithm"*, *Marcel Dekker Inc.*

Ford, D. M. 1985 "An analysis and application of a decoupled engine mount system for idle isolation". *SAE Technical Paper*, Paper No. 850976.

Iba, H.; H. deGaris; and T. Sato. 1996." A numerical Approach to Genetic Programming for System Identification", *Evolutionary Computation* 3(4): 417-452.

Ivakhnenko, A.G. 1971. "Polynomial Theory of Complex Systems", *IEEE Trans. Syst. Man & Cybern*, SMC-1, 364-378.

Kim, H. Y.; W.J. Bang and J.S. Kim. 1992 "Large deformation finite element analysis of automotive rubber components by using ABAQUS." *In: Proc ABAQUS user's conf.*, May 27-29, Newport, RI.

Kim, H. Y.; C. Choi; W. J. Bang; and J. S. Kim. 1993. "Large deformation finite element analysis for automotive rubber components" J. Korean Soc. Auto. Engng 15, 107-119.

Nariman-Zadeh, N.; A. Darvizeh; and R. Ahmad-Zadeh; 2003. "Hybrid Genetic Design of GMDH-Type Neural Networks Using Singular Value Decomposition for Modelling and Prediction of the Explosive Cutting Process", *Proceedings of the I MECH E Part B Journal of Engineering Manufacture*, Vol. 217, pp. 779 – 790.

Press W.H.; S. A. Teukolsky; W. T. Vetterling; and B. P. Flannery. 1992. "Numerical Recipes in FORTRAN: The Art of Scientific Computing", 2nd Edition, *Cambridge University Press*, (1992).

Rivlin, R. S. 1992 "The elasticity of rubber", *Rubber Chem. Technol.* 65, G51-G66.

Sanchez E.; T. Shibata; and L. A. Zadeh. 1997. "*Genetic Algorithms and Fuzzy Logic Systems*", World Scientific.

Yao, X. 1999. "Evolving Artificial Neural Networks", *Proceedings of IEEE*, 87(9):1423-1447.

Yu, Y.; G. Naganathan; and V. Dukkipati. 2001. "*A literature Review of Automotive Vehicle Engine Mounting Systems, Mechanism and Machine Theory*", Vol. 36, pp 123-142.

PERSON-CENTERED SIMULATION OF PRODUCT DEVELOPMENT PROCESSES USING TIMED STOCHASTIC COLOURED PETRI NETS

Torsten Licht
Lothar Dohmen
Peter Schmitz
Ludger Schmidt
Holger Luczak
Chair and Institute of Industrial Engineering and Ergonomics
RWTH Aachen University
D 52062, Aachen,
Germany
E-mail: {T.Licht; L.Dohmen; P.Schmitz, L.Schmidt, H.Luczak}@iaw.rwth-aachen.de

KEYWORDS
Research and Development, Corporate Planning, Decision Support Systems, Person-Centered Simulation, Petri Nets

ABSTRACT

Results of recent studies in the field of automotive industry affirm that productivity of product development departments provides room for tremendous improvements. Particularly work organization offers great potential in that respect. Simulation has proven to be an adequate technique to reveal potential for improving work organization e.g. in production planning or logistics.
Existing simulation approaches for *product development processes* however, are still lacking in maturity and therefore are not in practical use yet.
In this paper an executable simulation model is presented which goes beyond existing approaches enabling modeling and simulation of product development processes with respect to aspects of time, costs and quality.
The simulation model consists of five generic partial models which have been integrated into a consistent overall model: 1) model of products to be developed, 2) model of persons involved, 3) model of tools, 4) model of organizational structure and 5) model of product development projects.
The model was formally described by Timed Stochastic Coloured Petri Nets. Furthermore the model has been implemented with a standard simulation software. As first results selected data from verification tests as well as from validation by means of an empirical case study are presented.

INTRODUCTION

Improvements in the field of product development processes particularly can be expected from reducing weaknesses in project management. Major resources of shortcomings are delayed starting dates of projects or too tardily involving development partners. Long term goal of the research work described in this paper is a simulation based approach that enables project planners to early evaluate different organizational alternatives with respect to number of persons to assign, deadlines to affix for particular tasks etc. Basis for this evaluation could be several performance figures like project duration, idle periods or utilization of resources. Since project planners have to be able to participatorily evaluate such scenarios a simulation based approach has to fulfill several requirements such as conformity with user's expectations, self descriptiveness, suitability for individualization, error tolerance, controllability etc. Additionally all relevant quantities influencing duration of actions, capabilities of persons or availability of resources have to be representable. The well known formalism of Petri Nets enables participatory modeling of product development processes in that respect and was therefore used.

PRODUCT DEVELOPMENT PROCESSES

A *product development project* is a timely limited enterprise where variably abstract development targets of the management are transformed into concrete descriptions of new or changed products carried out by a project team (compare e.g. Pahl and Beitz 1999). This transformation process proceeds stepwise over several so called design phases where the description's degree of concretion increases. Typical design phases could be 1) system requirements, 2) conceptual design, 3) preliminary design, 4) detailed design, 5) integration and test. Thus, the core of *product development* is generation of knowledge with respect to attributes and capabilities of products. Therefore product development is a process of creative information processing. Beyond that, there are several characteristics of product development that limit the application of conventional, workflow based simulation techniques (e.g. Raupach 1999):

Iteration loops: During the product development process solutions of design problems are worked out, reviewed, evaluated and if necessary re-engineered. A poor solution can cause such an iteration loop as well as a deliberately stepwise proceeding where coarsely sketched versions are iteratively refined.

Non-deterministic operation: Due to uncertainty regarding boundary conditions and constraints duration and the course of product development processes can only be planned and

predicted poorly. Both structure of processes and details of single actions cannot be predicted satisfyingly.
Influence of tools: Tools – e.g. information and communication systems – used during product development can have considerable influence on the efficiency of a project. A simulation model has to be capable of mapping such effects.

Qualification of persons: Due to different levels of expertise and qualification bottlenecks in the development process can occur. That is, different members of a project team are differently capable of solving several problems.

Parallel processes: Usually involved persons are assigned to many development processes and beyond to several non-project related processes. Therefore a coordination of processes which compete for resources is needed. A major problem for planners of product development projects is the assessment of a new competing process influencing the duration of existing processes.

Inter-organizational cooperation: Even cooperation between different partners in the product development process one should be able to model. Particularly linkage of processes between different organizational units and companies by exchange of information should be object of simulation.

EXISTING SIMULATION APPROACHES

Due to this characteristics and the resulting requirements concerning simulation existing models and techniques from production planning and logistics cannot simply be adopted to product development.
One reason for that can be seen in the fact that these approaches focus on technical systems as well as logistical processes and human beings are solely considered as resources. Therefore these models are in fact automaton theoretic abstractions of the underlying production systems. System evaluation then is only based on objective measures.
An advanced simulation of a socio-technical system of work processes in manual manufacturing was described by Zülch et al. (e.g. 1998a, 1998b). In this approach structure of processes adapts to the tasks performed. Additionally aspects of qualification respectively restrictions due to lacking qualification were considered in this model. However, about modeling informatory-creative work processes – e.g. product development processes – this approach remains silent.
Regarding the discrete event simulation of work processes in engineering design the following approaches are worth mentioning: Levitt et al. (1994) developed a stochastic simulation approach – the so called Virtual Design Team. This approach was far-reaching in it's statements and capabilities. However, the simulation model presented in this paper goes beyond Levitt's approach: additionally to their model tasks, qualification profiles and preferences can be assigned to persons dynamically, furthermore persons can communicate with other involved persons. Therefore a concept of requests was implemented. Beyond, functional and geometrical interdependencies can be mapped on an abstract level by definition of so called interfaces between components.
Another stochastic approach for simulating product development processes was described by Raupach (1999). In this approach three partial models were distinguished: 1) the model of activities, 2) the model of resources and the organization as well as 3) the data model. A person in this model is considered as a resource of an organizational unit – and therefore has no active role.
With respect to person-centered simulation Steidel (1994) further developed the approach of Zülch and applied it to the domain of product development processes using a procedural queuing simulation software based on C. Because of this procedural structure the model is not suited for participatory develop a scenario with product developers. However, persons play an active role in this approach checking if a valid job is available and then processing this job in case of available resources. When more than one job is available persons decide which job to process first. However, priority rules for these decisions are based on a simple extension of the FIFO concept only. Persons in this approach do not have a specific goal system – only capabilities in the form of qualification describing parameters are mapped. Once an informatory-creative task is being performed by a person it will be completed continuously. Occasionally shifting attention between tasks to be performed concurrently is not possible. Furthermore in this approach tasks and the underlying components to be transformed are not explicitly distinguished.
A Petri Net based approach for modeling work processes in engineering design within the Concurrent Engineering concept was described by Kusiak and Yang (1993). This model distinguishes different generic steps of the design process. Besides the actual task sequence however, no further aspects of product development processes are considered. Furthermore only timeless Petri Nets are used, so that a dynamical simulation could not be performed.

PERSON-CENTERED SIMULATION MODEL OF PRODUCT DEVELOPMENT PROCESSES

The model presented in this paper basically consists of the five partial models: 1) model of products to be developed, 2) model of persons involved, 3) model of tools, 4) model of a hierarchical work organization and 5) model of product development projects. These partial models map different concepts of structures and processes. Subsequently these models are presented – as possible in the scope of this paper – and then integrated into an overall model.

Concept of the Model

Products to be developed are the actual work objects. Thus, products are object of a state transformation caused by actions of working persons.
Regarding simulation the focus of interest is on questions like which influences particular attributes of a product can have on: efforts spent for the transformation process, on duration of the transformation process or on product quality achieved by the transformation process. To assess these kinds of questions no "real" product data are used in the model but so-called "meta data" by which above mentioned influences can be mapped. In Figure 1 some of these data are

depicted. Basically products consist of different levels of aggregation which in turn can be presented in different levels of abstraction. The result is a consistent abstraction-decomposition-hierarchy of product views (e.g. Schmidt and Luczak 2000). The lowest aggregation level consists of so called *components*.

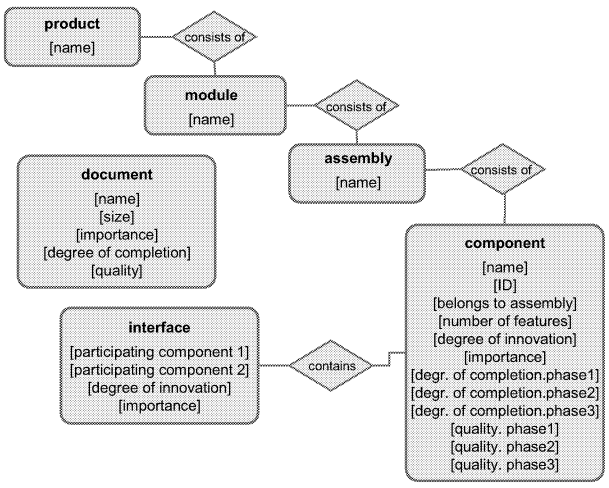


Figure 1: ER-Model of the Partial Model of Products

Each of these components is linked with other components by so called *interfaces* where interfaces can define several functional or geometrical interdependences. Two or more components can accumulate to assembly groups or modules in that way. Components can for instance be characterized by the number of included features or the degree of innovation. Interfaces can for instance be characterized by the involved components or the degree of innovation. The result of a product development process can also be a document like a FMEA. Althoght such documents are not directly part of the product they are considered in the partial model of products.

Working persons are the active elements in a work system i.e. the actual transformation process is carried out by persons. The *partial model of working persons* therefore basically speaks about matters of capabilities or qualifications and how powerful particular persons are. On the one hand can be described: what kinds of persons can perform what kinds of tasks on a qualitative level. On the other hand different degrees of efficiency of persons can be described on a quantitative level.
Persons in the model presented here play an *active role* in generating the system's dynamics – which is in contrast to the activity driven approaches from production planning and logistics. In terms of economic decision making and rational choice theories (e.g. Hill 2002) the simulated persons act according to the principles of a *homo oeconomicus*. Köhler et al. (2003) characterize homo oeconomicus as follows: 1) he is setting his owns goals to pursue, 2) he has resources at his disposal which can be limited, therefore he tries to use these resources with a maximal degree of utility, 3) marginal utility is diminishing with increasing activity spent on a task, therefore other tasks become more important in comparison, 4) pursuing a goal means not pursuing another goal – lost utility by not achieving the other goal is denoted as cost of opportunity; therefore homo oeconomicus compares costs of opportunity of different tasks to be performed, 5) each person has a specific goal system by which he can be distinguished from other persons – therefore assessing utilities takes place in a subjective fashion, 6) homo oeconomicus tends to up rate short term utilities – therefore long term consequences of an action can be disregarded easily. Please note here that these principles of homo economicus cannot be applied in entire strength. For instance a person can only "see" a small part of all available tasks to perform. He therefore cannot be able to absolutely rationally decide between two alternatives. The concept applied to the simulation model therefore was somewhat relaxed in terms of bounded rationality.
In the partial model of persons therefore persons assign priorities to differed tasks to be performed according to a personal goal system and several boundary conditions.
Based on these priorities persons define their own sequence of tasks as well as a short term distribution of attention with respect to different product development projects. The act of assessing priorities takes place on a regular basis and includes the task actually being performed. Thus, it is possible that tasks will only be processed partially since the person changes between different tasks. In doing so a system's dynamic unfolds that corresponds to common sense observations in the field of engineering design or in other domains where informatory-creative forms of work predominate (e.g. in the scientific community).
Basically persons can perform *managerial* tasks and *technical* tasks. Managerial tasks can for instance be delegation of work, evaluation of work done, redirecting work done to other persons if evaluated as good or redirecting work to the executing person (initiate an iteration loop) if quality is poor. Different levels of experience of managerial workers and their influences on performance of managerial tasks are not yet mapped in the latest model.
Performance of technical tasks (e.g. analyzing requirements, gathering information, transform product description, computing etc.) however, can differ with respect to efficiency as a result of specific occupational careers and hereby gathered experiences. Additionally, short term variation of efficiency is mapped by person-specific probability distributions.

Tools can have a tremendous impact on effort, duration and quality of the work system's output. There even are tasks that can be processed only by using tools. Tools can be physical devices like paper and pencil or computational utilities like different CAD systems one the one hand. But tools can also be models, methods like creativity techniques or methodologies like the design methodology described by Pahl and Beitz (1999) on the other hand.
In the scope of the simulation model however, only tools are considered and modeled that are potential bottleneck resources. A specification of what tools can be used for what tasks is done in the partial model of project development projects. The actual allocation of particular tools in turn, is dynamically done by persons.

The *organizational structure* of a product development project generally is based on the overall organizational structure of the company (Pahl and Beitz 1999). However, in

recent years product development increasingly takes place in so-called concurrent engineering teams (Bullinger and Warschat 1995). In these teams parallel and integrated definition of product descriptions and production processes is carried out. Therefore these concurrent engineering teams usually consist of members of different organizational units or even companies. One team member usually is involved in two or more parallel projects. The *partial model of organizational structure* as used in the simulation model is illustrated in Figure 2.

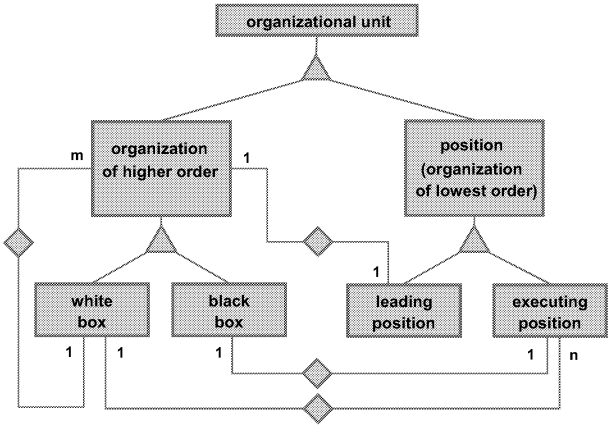


Figure 2: Organizational structure of the simulated product development team

Positions as smallest possible organizational units are defined by assigning tasks, possible resources and restrictions to particular persons. Positions can be distinguished in *managerial* positions and *technical* positions (compare e.g. Luczak 1998). Furthermore positions can form organizational units of higher order. These units can be denoted as *white box* units if the respective inner structure is explicitly modeled. If the inner structure is not modeled, one may speak of a *black box* unit which then can be characterized by its properties and capabilities from an outside point of view only. Persons who are assigned to a particular position of a white box unit can for instance communicate with other persons or with black box units.

The *operational structure* (as opposed to the organizational structure) defines work processes and orders of work processes which are necessary to perform tasks. Thus, operational structure defines content, extent and specific requirements for each task (Luczak 1998). Regarding the concept of the simulation model operational structure has been included into the *partial model of development projects*.
Within the above mentioned design phases a project can be decomposed into single *tasks*. In computational terms tasks can be considered as pointers to the work to be done, i.e. components to be transformed, positions to transform, adequate tools and resources etc. can be specified here. Furthermore it is possible to assign a deadline to a task. This deadline to some extend influences the priority the person assigns to a task. Importance of the task for organizational or personal goals can also be defined. Serial, parallel or hybrid forms of task structures can be defined by assigning respective predecessor-successor-relations. This definition can be carried out in a so-called *task net*.
Actual design phases can be mapped by definition of milestones. These milestones are points of time where the actual status of the project is compared to a target status. The project leader then has to decide if the project should be pursued or not. Such a withdrawal decision is not implemented yet in the computational simulation model.
However, in case of not fulfilling a defined target quality of the product description a project leader can reassign the task to the respective person to increase quality. In order to be able to start proceeding with a task of a subsequent phase all tasks of the previous phase have to be completed. Together with the mentioned concept of quality assessment so-called *quality gates* are realized in the simulation model.

The five partial models described basically denote *structures* of product development processes. In the scope of integrating these partial models into an overall model single structural elements were linked in the sense of a flow model. The result is a process model in terms of system's theory with inputs, outputs, elements with interdependent relations and transformation relations. Therefore identifying and configuration of these links was crucial.

Implementation of the Model

To formally describe the simulation model the notion of Timed Stochastic Coloured Petri Nets was consequently used. To do so all aspects of the overall model were mapped into a directed graph consisting of places, transitions, arcs and markings. Specific starting and boundary conditions are depicted by respective properties of the model's elements (Zhou and Venkatesh 1999). On the one hand the graphical representation of the model leads to transparency, consistency and conformity with user's expectations so that model generation and maintenance can be performed in a participatory way. On the other hand maturity of the Petri Net notion allows a direct implementation into computer models where dynamics of variables of interest can be studied.

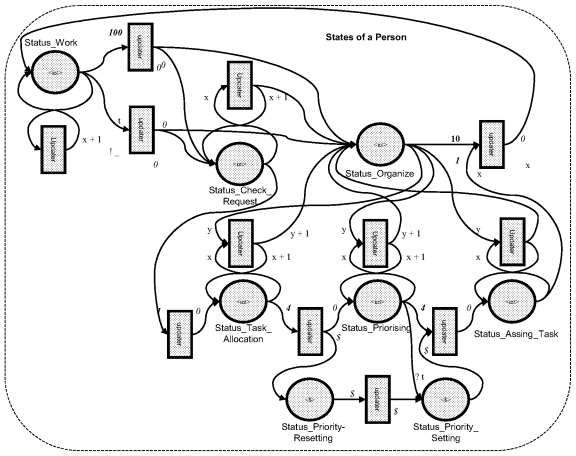


Figure 3: Main of States of a Person

Subsequently implementation of selected concepts are presented which mainly define the person-centered character as well as the resulting system's dynamic that unfolds.
The main states of a person as depicted in Figure 3 like organizing (task allocation, processing or generating requests, priority setting etc.) or working (adjust to a task, performing basic activities like information gathering, transforming product description, etc.) are passed through sequentially and on a regular basis. Time slices for this sequences can be adjusted dynamically during runtime.

Figure 4 depicts the concept of *task allocation* in the notion of Petri Nets. By showing two persons here it was illustrated that this concept can be expanded arbitrarily.

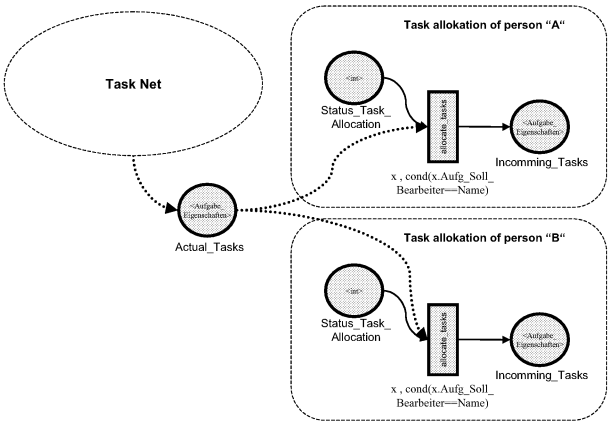


Figure 4: Task Allocation of Two Persons

One of the characteristics of product development lies in problems in terms of lacking information occurring during processing tasks. In such a case a product developer somehow has to gather this information. Often this is done by sending queries to other persons who might be able to help. While waiting for that particular information the product developer may process another task. The concept of processing requests was modeled according to the Petri Net in Figure 5.

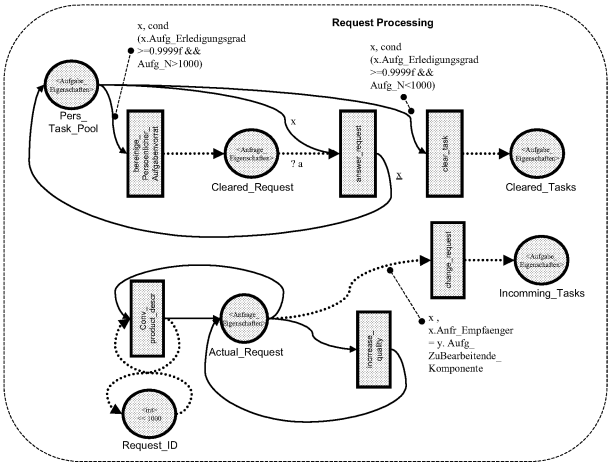


Figure 5: The Concept of Processing Requests

Somewhat more complex is the concept of *priority setting* as depicted in Figure 6. Persons decide on a regular basis with respect to their personal goal system which task to spend the own attention and power next. The mechanism for that works as follows: on a regular basis the values of priorities of each task in the own `Pers_Task_Pool` is reset to be able to afresh all priorities then.
Generally a person trades off costs and utilities in terms of bounded rationality. The task with a high value of expected utility will be assigned with a high priority. Utility depends on the personal goal systems and the chance to be able to complete a task by time which includes the efforts to be spent and the time still available to complete the task.

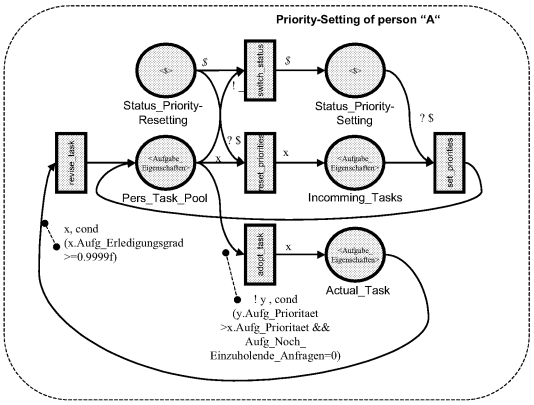


Figure 6: Priority Setting of a Person

Simultaneously the expected costs of transaction in terms of adjustment to a new task are assessed. Let π_i be the calculated *priorities* of tasks *i* in the `Pers_Task_Pool` and c_i the additional *costs of transaction*. Then transition `adopt_task{}` in Figure 6 can fire for task *i* if

$$\max(\pi_i - c_i)$$

and if there is no "open request" for this task receivable. Costs of transaction c_i for performing a task increases if the task has not been performed a longer time. The so-called *actual setup time* t_{AST} corresponds to that concept. Let *a* be the *activity* that corresponds to the frequency of proceeding a task in a recent time span. Let furthermore T_{ST} be the *absolute setup time* for firstly adjust to a task and let T_{exp} be the *expected time* effectively spent for completing a task from the beginning to the end (not including idle times). Then t_{AST} is

$$t_{AST}(a) = (1 - \frac{a}{T_{\exp}}) \cdot T_{ST}$$

With a *personal weight* K_{ST} which is part of the personal goal system of a person costs of transaction c_i for performing a task result in

$$c_i = t_{AST\,i} \cdot K_{ST}$$

Let n_t be the *number of tasks* which are situated in `Pers_Task_Pool` to a time point of evaluation *t*. Assume that a person's strategy is to *evenly* promote all task. The *available time* t_{av} for performing a task until d*eadline* T_{dead} will be reached then is denoted as

$$t_{av} = \frac{T_{dead} - t}{n_t}$$

Let now δ_c be the *degree of completion* of a task. The *required time* t_{req} for completing the task then results in

$$t_{req} = T_{\exp} \cdot (1 - \delta_c)$$

Let *u(t)* be the *urgency* of a task, then

$$u(t) = \begin{cases} \frac{t_{req}}{b}, (T_{dead} - t) \le 0 \\ \frac{t_{req}}{b + t_{av}}, (T_{dead} - t) > 0 \end{cases}$$

The limiter *b* prevents *u(t)* from reaching inadmissible values for $\lim_{t_{av} \to 0} u(t)$. Let furthermore be I_c the *importance* of a task for the *company's goals* and I_p be the *importance* of a task for *personal goals*. With the respective personal weights the *importance I* of a task *i* yields

$$I_i = I_{ci} \cdot K_c + I_{p_i} \cdot K_p$$

The priority π_i finally results in

$$\pi_i(t) = \frac{u_i(t) \cdot I_i}{a_i \cdot K_a}$$

where K_a is a personal weight which expresses to which extent a person tries to evenly promote all tasks.
The transition `set_priorities{}` regularly assesses all tasks in `Pers_Task_Pool`, `Incomming_Tasks` and in `Actual_Tasks`. Therefore priorities are never absolute but always relative to the priorities of other tasks to be performed.

FIRST RESULTS AND DISCUSSION

The Petri Net of the simulation model was verified with respect to boundedness, safety, blocking as well as conservation and then implemented using the Petri Net-Simulator "Poses++" (http://www.gpc.de) which has proven to be convenient for implementing large scale models. At present the model consist of 16 modules where alone the partial model of persons consists of 6 modules. By linking the modules in terms of object orientation one single consistent Petri Net of the overall model emerges.
With respect to the verification of the simulation model especially adjustment of the high number of interdependent parameters is crucial. These parameters create very complex system's dynamics which in turn makes an examination and evaluation of the model difficult.
First verification runs however, implicated that the system largely behaved as expected.

The unfolding system's dynamic is depicted in Figure 7 where 6 task with identical values for importance *I* were performed in parallel by one person. Bars in the line of a task indicate that this particular task has been performed to some degree during that time span. Note that all of the above mentioned parameters of all tasks have been kept constant. So differences in overall duration of performing a task obviously are founded in interacting of tasks and parameters of decision making.

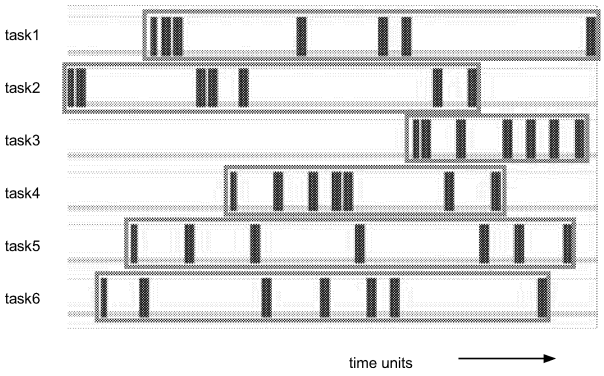


Figure 7: One Person Performing 6 Tasks

In terms of the *ceteris paribus* concept all parameters were kept constant and only parameters of interest were changed to assess plausibility of the parameter's impact on the system's behavior. During these first runs the following concepts have been ignored in order to keep the system's dynamics roughly in a manageable scope: concepts of sending requests (with disabling the respective task until the answer was received), of overlapping design phases, of redirecting poor-quality results, of dynamically assigning tasks or time slices or of dynamically changing experience.
In terms of plausibility the influence of the following parameters on system's dynamics have been assessed in various pre-tests: number of persons, number of tasks and underlying components, T_{exp}, T_{ST}, T_{dead}, I_p, I_c, K_p, K_c, K_a and *b*.
Characteristics of degrees of completion and of urgency of two tasks are illustrated in Figure 8 and 9. Task1 and task2 are processed nearly evenly.

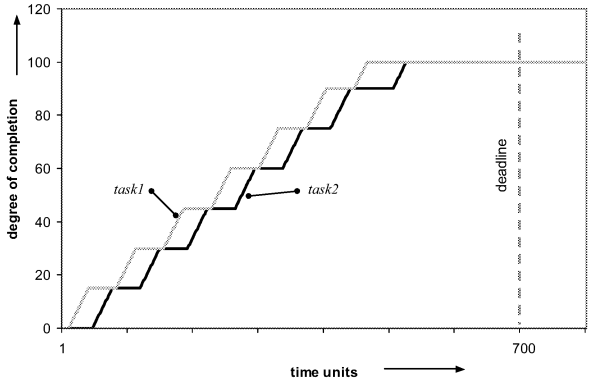


Figure 8: δ_c - Two Tasks Evenly Performed

The belonging characteristics of the respective urgencies of each task are illustrated in Figure 9. It can be found that urgency of task1 decreases as soon as task1 is being performed. The value of urgency of task2 however is increasing simultaneously. The high levels of urgency left when both tasks are finished can be explained by persons who do not reassess urgency when a task is completed. Therefore the last value keeps "saved".

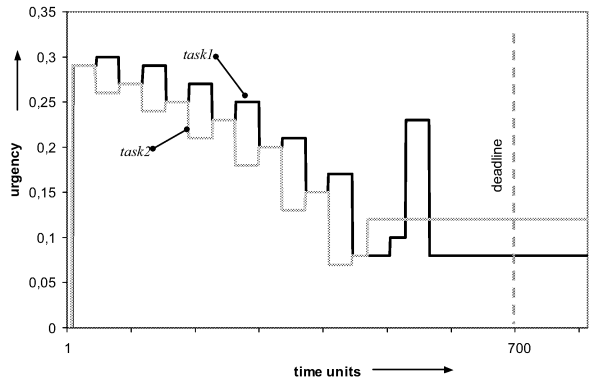


Figure 9: Respective Characteristics of Urgency *u(t)*

With this adjustments now various studies regarding the tendential influence of particular parameters with respect to system's dynamic could be done. If for instance the importance I_c of task1 for the company's goals was tremendously increased, characteristics considerably changed. In Figure 10 and 11 the results are illustrated: The working person does not start processing task2 even thought task2's value for urgency notably increases as the deadline comes in sight.

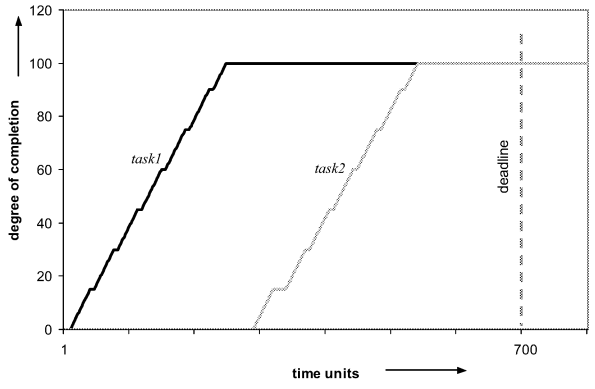


Figure 10: δ_c - Two Parallel Tasks with $Ic_1 >> Ic_2$

In order to verify the prototypically implemented simulation model with respect to the conceptual model a large number of experiments has been conducted. In these experiments parameters of the model have been varied systematically. Oneway ANOVAs were used to prove effects of parameter variation on the output variables of interest.

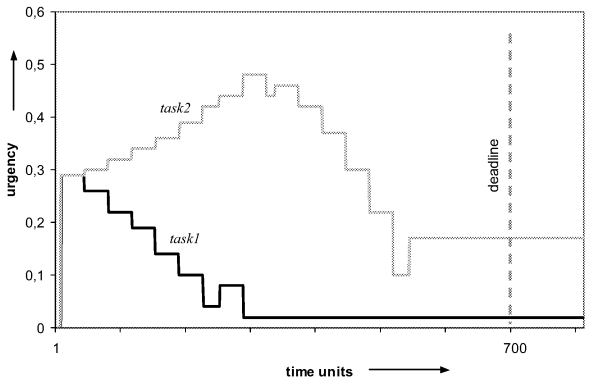


Figure 11: Urgency *u(t)* of Two Parallel Tasks with $I_1 >> I_2$

As a result of these experiments some minor shortcomings could be identified – and consequently eliminated. For instance the mechanism for assigning priorities to tasks had to be improved. In an early version of the implementation a person was not able to assess more than a certain number of tasks within one time slice. After improving the model time slices now are assigned dynamically so a person can assess as much tasks as necessary.

As a next step the simulation model was validated with respect to an empirical case study. Therefore, in a small-sized company of the german electronics industry a product development project was surveyed. Data relevant for all partial models were collected and implemented in a simulation model of the project.
For a test-statistical validation of the simulation output two of the tasks have been exemplarily chosen: 1) the cycle time of the "conceptual design" and 2) the cycle time of the "layout design". Additionally, the total cycle time was considered as an output variable. Therefore, these output variables have been tested against the empirically collected variables – which are available as single values (no distribution). The most sensitive input variables were the time effort T_{exp} for each task which are also available as single values with a precision of 30 minutes ($T_{exp\ total}$ = 237 h, $T_{exp\ concept}$ = 130 h, $T_{exp\ layout}$ = 45 h).
In order to be able to use t-tests the simulated output variables have to be independent and normal distributed. Since terminating simulations (instead of steady-state) were used the output variables were independent by definition.

Table 1: Statistics of the three considered output variables and one-sample t-tests against the empirical values

	N	mean [d]	sd [d]	standard error of means	empirical value [d]	T	df	sig. (2-sided)	mean of differences	95% confidence intervall of differences lower	upper
total cycle time	200	69.3	9.25	.654	69	.420	199	.675	.275	-1.02	1.56
cycle time „conceptual design"	200	66.1	10.0	.708	66	.177	199	.860	.125	-1.27	1.52
cycle time "layout design"	200	18.9	4.70	.332	22	9.419	199	<.001**	-3.13	-3.79	-2.47

Normality of distributions was assured by using Kolmogorow-Smirnow tests. With $N = 200$ simulation runs none of the output variables were significantly different from a normal distribution.
In Table 1 results of one-sample t-tests of the output variables against the single empirical values are shown. Neither the total cycle time of the simulated product development project nor the cycle time of the "conceptual design" were significantly different from the empirically collected values. Merely the simulated cycle time of "layout design" was significantly different from the empirical value. Acknowledging the fact that in real product development projects cycle times are distributed with a much higher variance than in the one-sample t-test (variance is neglected here) this result can be relaxed. If additionally the relatively small time effort of "layout design" is considered the simulated cycle time of 18,9 days still is acceptably close to the empirically collected 22 days.
These first results and, additionally, some qualitative characteristics (according to Figures 8 and 10) of the degrees of completion of tasks of the surveyed product development project have been assessed by some experts of the electronics company. In an expert interview the results were validated as being realistic.

FUTURE RESEARCH

The model presented in this paper contains elements of some well known approaches. In integrating many aspects of these different approaches the new model speaks about matters where the others remain silent. For instance regarding a realistic system's dynamics the model presented so far seems to exceed capabilities of the other approaches. Improving possibilities for parametrizing the model (up to now this is done at code level) as well as for advanced, automated analysis of simulation results the model will be able to support managers of product development processes. Particularly the modular structure and the partial models which properly map product development processes enable planners of such projects to participatorily "play around" with different organizational alternatives and assess the results.
Further research has to be done to systematically verify the simulation model presented in this paper. So far values for priorities of tasks alternate between very large ranges. Therefore ratios between several priority influencing parameters have to be balanced and adjusted in order of magnitude. Also ratios between several time slices in conjunction with costs of transaction have to be balanced and fit to empirical findings. Especially the variances of task's cycle times have to be surveyed empirically to better be able to validate simulation models.
From a human factors point of view it would be very interesting to assign well know findings with respect to patterns of behavior to some of the parameters. It could for instance be realized that one person works very fast but produces a low quality work result while another person always reaches excellent quality but never completes his work in time.

ACKNOWLEDGEMENTS

This research was funded by the German Research Foundation (Deutsche Forschungsgemeinschaft), grant number Lu 373/26-1.

REFERENCES

Bullinger, H.-J. and J. Warschat. 1995. "Concurrent Simultaneous Engineering Systems. The Way to Successful Product Development." Springer-Verlag, Berlin.

Hill, P.B. 2002. "Rational-Choice-Theorie". Transcript-Verlag, Aachen.

Kusiak, A. and H.H. Yang. 1993. "Modeling the Design Process with Petri Nets". In Concurrent Engineering 1993, H. Parsei and W.G. Sullivan (Eds.). Chapman & Hall, London.

Köhler, M.; Mold, D.; Rölke, H.; Spresny, D. 2003. „Handlung und Struktur. Modellierung von Akteurmodellen". In *Sozionik – Modellierung soziologischer Theorie*, R.v. Lüde; D. Mold; R. Valk (Eds.). Lit Verlag, Münster.

Levitt, R.E.; G.P. Cohen; J.C. Kuntz; C.I. Nass; T. Christiansen; and Y. Jin. 1994. "The Virtual Design Team: Simulating How Organizational Structure and Information Processing Tools Affect Team Performance." In Computational Organization Theory 1994, K.M. Carley and M.J. Prietula (Eds.). Lawrence Erlbaum Assoc., Hillsdale, N.J.

Luczak, H. 1998. "Arbeitswissenschaft", 2. Auflage. Springer-Verlag, Berlin.

Pahl, G. and W. Beitz. 1999. "Engineering Design. A Systematic Approach." 2nd Ed. Springer-Verlag, Berlin.

Raupach, H.-C. 1999. "Simulation von Produktentwicklungs-prozessen." Dissertation, TU Berlin, Berlin

Schmidt, L. and H. Luczak. 2000. "Knowledge Representation for Engineering Design Based on a Cognitive Model." In Proceedings of the IEA 2000/ HFES 2000 Congress (San Diego, CA, Jul.29-Aug.4). Human Factors and Ergonomics Society, 1/623-1/626.

Steidel, F. 1994. "Modellierung arbeitsteilig ausgeführter, rechnerunterstützter Konstruktionsarbeit – Möglichkeiten und Grenzen personenzentrierter Simulation." Dissertation, TU Berlin, Berlin.

Zhou, M and K. Venkatesh. 1999. "Modeling, Simulation, and Control of Flexible Manufacturing Systems – A Petri Net Approach." World Scientific, Singapore.

Zülch, G.; J. Krüger; and H. Schindele. 1998a. "Planung qualitätsförderlicher Personalstrukturen im Fertigungsbereich." Zeitschrift für Arbeitswissenschaft 52, No.1, 22-29.

Zülch, G.; J. Heel; and B. Brinkmeier. 1998b. "Personalorientierte Simulation als Hilfsmittel zur Strukturierung von Arbeitsaufgaben." Zeitschrift für Arbeitswissenschaft 52, No.3, 176-184.

QUEUEING NETWORKS MODELS OF OUT-PATIENT CLINICS AND HOSPITALS

Bogusław Filipowicz and Joanna Kwiecień
Department of Automatics
AGH University of Science and Technology
Al. Mickiewicza 30, 30-059 Kraków
Poland
E-mail: filip@ia.agh.edu.pl, kwiecien@ia.agh.edu.pl

ABSTRACT

This article shows how open and closed queueing networks helped to analyse a health center operation. The mode of the clinic operation has been discussed and mathematical models of queueing network have been proposed. Performance measures that describe the queueing network operation have been obtained. These parameters have been employed in assessment of the clinic efficacy. The queueing networks analysis showed which nodes were the bottleneck.

INTRODUCTION

Queueing networks consisting of several nodes (offering the service) and customers (requiring the service) better represent the structure of many systems than single service stations. Depending on the total number of customers queueing networks can be categorized as: open, closed and mixed. For example the open queueing networks can receive customers from outside to receive services at network's nodes. A queueing network is closed if there is a fixed number of customers. Depending on the total number of customer classes there are single class networks and multiple class networks. Queueing networks are described by the number of nodes, number of job classes and the type of node. The simpliest queueing networks are Jackson networks, which consist of only one job class and Markov systems (service times are exponentially distributed, the arrival process is Poisson), but the most common are BCMP networks.

Queueing networks represent a very important application of queueing theory. They are commonly used in the area of communication systems, flexible manufacturing and computer science (Bolch et al. 1998; Czachórski 1999; Kleinrock 1976). Queueing theory has also been used extensively for the modelling of many service stations, for example in the health care. There are some recent studies in the health care area (Albin et al. 1990; Doi et al. 1997; Filipowicz 1997; Kwiecień 2004).

In this paper we present the application of multiclass queueing networks in health centers operations. We use open and closed BCMP networks for analysing our "real" models. We also include priority queueing. Queueing networks models and methods to calculate all the performance measures of out-patient clinic and hospital are described in the following sections.

BACKGROUND

The type of queueing network which is easy to evaluate is called product form or separable. The BCMP queueing network is a network discussed by Baskett, Chandy, Muntz and Palacios (Baskett et al. 1975; Bolch et al. 1998; Czachórski 1999; Flipowicz 1997) to which we can apply the BCMP theorem. These networks include several job (customer) classes, different queueing discipline (FIFO – first input first output, LIFO-PR – last input first output with preemptive, PS – processor sharing, IS – infinite server) and generally distributed service times. The customer classes can differ in their routing probabilities, a customer can change its class while passing through the network. These assumptions give us the four node types:

- Type 1: the service times are exponentially distributed and for different customer classes must be identical, the service discipline is FIFO, node with multiple servers;
- Type 2: different customer classes have different general service time distribution, node with one server, the service discipline is PS;
- Type 3: different customer classes have different general service time distribution, node with an ample number of servers (there is no queue);
- Type 4: different customer classes have different general service time distribution, node with one server, the service discipline is LIFO-PR.

For Type 2, Type 3 and Type 4 nodes, the service times of the customers of different classes must have a rational Laplace transform. To describe queueing networks and the determined performance measure we use the following symbols and equations:

- N – number of nodes
- R – number of classes
- m_i – number of servers of the ith node
- μ_{ir} – the service rate of the rth class in the ith node
- $p_{js,ir}$ – the routing probability (the probability that a customer of the sth class at the jth node is transferred to the customer of the rth class of the ith node)
- $p_{0,ir}$ – the probability that a customer entering the network from outside enters the ith node as a customer of the rth class
- ρ_{ir} - the utilization of the node with index i by the rth class is given by:

$$\rho_{ir} = \frac{\lambda_{ir}}{m_i \mu_{ir}} \quad \text{for } i = 1,...,N;\ r = 1,..,R \qquad (1)$$

- e_{ir} – the mean value of visits of the rth class to the ith node:

$$e_{ir} = p_{0,ir} + \sum_{j=1}^{N}\sum_{s=1}^{R} e_{js}p_{js,ir} \quad \text{for open network} \tag{2}$$

$$e_{ir} = \sum_{j=1}^{N}\sum_{s=1}^{R} e_{js}p_{js,ir} \quad \text{for closed network} \tag{3}$$

- λ_{ir} – the throughput of customers of the rth class in the ith node is calculated as follows:

$$\lambda_{ir} = \lambda_{0,ir} + \sum_{j=1}^{N}\sum_{s=1}^{R} \lambda_{js}p_{js,ir} \quad \text{for open network} \tag{4}$$

$$\lambda_{ir} = \sum_{j=1}^{N}\sum_{s=1}^{R} \lambda_{js}p_{js,ir} \quad \text{for closed network} \tag{5}$$

- λ_r – the overall throughput can be written as:

$$\lambda_r = \sum_{i=1}^{N} \lambda_{0,ir} \quad \text{for open network} \tag{6}$$

$$\lambda_r = \frac{\lambda_{ir}}{e_{ir}} \quad \text{for closed network} \tag{7}$$

A closed multiclass queueing network is much more difficult to solve. In the literature there are numerous algorithms to calculate the performance measures, such as the mean value analysis (Bolch et al. 1998; Reiser and Lavenberg 1980), the convolution algorithm (Buzen 1973), the flow equivalent server method (Bolch et al. 1998). In this paper we use the summation method (Bolch et al. 1998), which is based on the idea that the mean number of customers at each node is a function of throughput f_{ir} given by:

$$f_{ir}(\lambda_{ir}) = \overline{K}_{ir} = \begin{cases} \dfrac{\rho_{ir}}{1-\dfrac{K-1}{K}\rho_i}, & Type 1,2,4 (m_i = 1) \\ m_i\rho_{ir} + \dfrac{\rho_{ir}}{1-\dfrac{K-m_i-1}{K-m_i}\rho_i}\cdot P_{m_i}, & Type 1 (m_i > 1) \\ \dfrac{\lambda_{ir}}{\mu_{ir}}, & Type\ 3 \end{cases} \tag{8}$$

with:

$$\rho_i = \sum_{r=1}^{R}\rho_{ir}; \quad K = \sum_{r=1}^{R} K_r$$

$$P_{mi} = \frac{(m_i\rho_i)^{m_i}}{m_i!(1-\rho_i)} \cdot \frac{1}{\sum_{k_i=0}^{m_i-1}\dfrac{(m_i\rho_i)^{k_i}}{k_i!} + \dfrac{(m_i\rho_i)^{m_i}}{m_i!}\cdot\dfrac{1}{1-\rho}}$$

We can obtain with $\lambda_{ir} = \lambda_r e_{ir}$ the following expression:

$$\sum_{i=1}^{N} f_{ir}(\lambda_r e_{ir}) = K_r, \quad \text{for } r = 1,...,R$$

An algorithm for multiclass queueing networks which determine λ_r can be described in the following steps (Bolch et al. 1998):

1. Initialization: $\lambda_1 = \lambda_2 = ... = \lambda_R = 0.00001$, $\varepsilon = 0.00001$
2. Computation λ_r for all r = 1,..,R according to:

$$\lambda_r = \frac{K_r}{\sum_{i=1}^{N} fix_{ir}(\lambda_1,...,\lambda_R)} \tag{9}$$

with:

$$fix_{ir} = \begin{cases} \dfrac{\dfrac{e_{ir}}{\mu_{ir}}}{1-\dfrac{K-1}{K}\rho_i}, & Type 1,2,4 (m_i = 1) \\ \dfrac{e_{ir}}{\mu_{ir}} + \dfrac{\dfrac{e_{ir}}{m_i\mu_{ir}}}{1-\dfrac{K-m_i-1}{K-m_i}\rho_i}\cdot P_{m_i}(\rho_i), & Type 1 (m_i > 1) \\ \dfrac{e_{ir}}{\mu_{ir}}, & Type\ 3 \end{cases}$$

3. Determine error:

$$e = \sqrt{\sum_{r=1}^{R}(\lambda_{r_n} - \lambda_{r_{n+1}})^2}$$

   if e > ε go back to Step 2.
4. Compute the performance measures of the network.

In the paper we also consider queueing networks with priority classes. For the analysis of such priority networks we use the extended summation method so-called PRIOSUM (Bolch et al. 1998). We restrict our attention to the queueing consisting of the M/M/m/FIFO/∞ HOL nodes (with non preemptive head-of-line) and assume that class 1 has the highest priority and class R has the lowest priority. In this case we obtain:

$$\overline{K}_{ir} = m_i\rho_{ir} + \frac{\lambda_{ir}\sum_{j=1}^{R}\dfrac{\rho_{ij}}{\mu_{ij}}}{\left(1-\dfrac{K-m_i-1}{K-m_i}\delta_{ir}\right)\left(1-\dfrac{K-m_i-1}{K-m_i}\delta_{ir+1}\right)}\cdot\frac{P_{m_i}(\rho_i)}{m_i\rho_i} \tag{10}$$

$$fix_{ir} = \frac{e_{ir}}{\mu_{ir}} + \frac{e_{ir}\sum_{j=1}^{R}\dfrac{\rho_{ij}}{\mu_{ij}}}{\left(1-\dfrac{K-m_i-1}{K-m_i}\delta_{ir}\right)\left(1-\dfrac{K-m_i-1}{K-m_i}\delta_{ir+1}\right)}\cdot\frac{P_{m_i}(\rho_i)}{m_i\rho_i} \tag{11}$$

δ_{ir} denotes the sum of the utilization of the ith node with respect to customers of classes with priority higher than rth

class, δ_{ir+1} denotes the sum of the utilization of the ith node with respect to customers of classes with a higher or equal priority.

QUEUEING MODELS OF CLINIC OPERATIONS

This section presents three applications of multiclass queueing networks in the health care area. Consider one community health center and a chemotherapy unit.

Example 1

Our real community health center consists of six out-patient clinics modeled as M/M/m/FIFO/∞ systems (Poisson arrival, exponential distribution of the service times, m is the number of doctors):

- dermatological out-patient clinic S_1: one dermatologist, exponentially distributed service time with mean $1/\mu_1$ = 8,3 minute, Type 1 according to the BCMP classification;
- gynaecological out-patient clinic S_2: one gynaecologist, exponentially distributed service time with mean $1/\mu_2$ = 7,25 minute, Type 1;
- ophthalmological out-patient clinic S_3: one ophthalmologist, exponentially distributed service time with mean $1/\mu_3$ = 6,7 minute, Type 1;
- otolaryngological out-patient clinic S_4: two otolaryngologists, exponentially distributed service time with mean $1/\mu_4$ = 7,2 minute, Type 1;
- surgical out-patient clinic S_5: one surgeon, exponentially distributed service time with mean $1/\mu_5$ = 7,7 minute, Type 1;
- cardiological out-patient clinic S_6: one cardiologist, exponentially distributed service time with mean $1/\mu_6$ = 11 minutes, Type 1.

We assume that service rates are load independent in each doctor's office. Now we introduce two different classes of patients:

- class „1" – coming and treatment patients in our clinic;
- class „2" – patients who should continue the treatment in another clinic or hospital.

The matrix shown in Table 1 describes finite-state Markov chain whose states are node-class pairs.

The routing matrix is not reduced so that we obtain performance measures without the concept of chains (Bolch et al. 1998).

According to Eq. (4) the throughputs at each node by class type are given:

$$\lambda_{11} = 0{,}1223\lambda;\ \lambda_{12} = 0{,}0058\lambda;\ \lambda_{21} = 0{,}1227\lambda;$$
$$\lambda_{22} = 0{,}0173\lambda;\ \lambda_{31} = 0{,}1573\lambda;\ \lambda_{32} = 0{,}0173\lambda;$$
$$\lambda_{41} = 0{,}2886\lambda;\ \lambda_{51} = 0{,}1775\lambda;\ \lambda_{52} = 0{,}0115\lambda;$$
$$\lambda_{61} = 0{,}0851\lambda..$$

To obtain the mean number of visits, Eq.(2) is used to get:

$$e_{11} = 0{,}1290;\ e_{12} = 0{,}1111;\ e_{21} = 0{,}1294;\ e_{22} = 0{,}3333;$$
$$e_{31} = 0{,}1659;\ e_{32} = 0{,}3333;\ e_{41} = 0{,}3044;\ e_{51} = 0{,}1872;$$
$$e_{52} = 0{,}2223;\ e_{61} = 0{,}0898.$$

Because $\lambda_1 = 0{,}9481\lambda$ and $\lambda_2 = 0{,}0519\lambda$ so that the utilization of each node by class type is:

$$\rho_{11} = \frac{0{,}1223\lambda}{\mu_1};\ \rho_{12} = \frac{0{,}0058\lambda}{\mu_1};\ \rho_{21} = \frac{0{,}1227\lambda}{\mu_2};\ \rho_{22} = \frac{0{,}0173\lambda}{\mu_2};$$
$$\rho_{31} = \frac{0{,}1573\lambda}{\mu_3};\ \rho_{32} = \frac{0{,}0173\lambda}{\mu_3};\ \rho_{41} = \frac{0{,}2886\lambda}{2\mu_4};$$
$$\rho_{51} = \frac{0{,}1775\lambda}{\mu_5};\ \rho_{52} = \frac{0{,}0115\lambda}{\mu_5};\ \rho_{61} = \frac{0{,}0851\lambda}{\mu_6}.$$

For the M/M/1/FIFO/∞ and the M/M/2/FIFO nodes the mean number of patients by class type (K_{ir}) can be computed from the well-known formula:

$$\overline{K}_{irM/M/1/FIFO/\infty} = \frac{\rho_{ir}}{1-\rho_i};$$
$$\overline{K}_{irM/M/2/FIFO/\infty} = \frac{2\rho_{ir}}{1-\rho_i^2}$$

The mean response times T_{ir} can be calculate using Little's theorem ($T_{ir} = K_{ir}/\lambda_{ir}$).

Table 1: The Routing Matrix (i – node number, r – class index)

(i;r)	(0;1)	(0;2)	(1;1)	(1;2)	(2;1)	(2;2)	(3;1)	(3;2)	(4;1)	(5;1)	(5;2)	(6;1)	(S_{wy};1)	(S_{wy};2)
(0;1)	0	0	0,1233	0	0,1294	0	0,1659	0	0,3044	0,1872	0	0,0898	0	0
(0;2)	0	0	0	0,1111	0	0,3333	0	0,3333	0	0	0,2223	0	0	0
(1;1)	0	0	0	0	0	0	0	0	0	0	0	0	1	0
(1;2)	0	0	0	0	0	0	0	0	0	0	0	0	0	1
(2;1)	0	0	0	0	0	0	0	0	0	0	0	0	1	0
(2;2)	0	0	0	0	0	0	0	0	0	0	0	0	0	1
(3;1)	0	0	0	0	0	0	0	0	0	0	0	0	1	0
(3;2)	0	0	0	0	0	0	0	0	0	0	0	0	0	1
(4;1)	0	0	0	0	0	0	0	0	0	0	0	0	1	0
(5;1)	0	0	0,0305	0	0	0	0	0	0	0	0	0	0,9695	0
(5;2)	0	0	0	0	0	0	0	0	0	0	0	0	0	1
(6;1)	0	0	0	0	0	0	0	0	0	0	0	0	1	0
(S_{wy};1)	1	0	0	0	0	0	0	0	0	0	0	0	0	0
(S_{wy};2)	0	1	0	0	0	0	0	0	0	0	0	0	0	0

We finally have performance measures in Table 2.

Table 2: Mean Number of Patients and Mean Response Time in Our Clinic

	S_1	S_2	S_3	S_4	S_5	S_6
K_{i1}	2,11	1,59	2,72	2,43	14,24	1,57
K_{i2}	0,10	0,23	0,30	-	0,93	-
$T_{i1}=T_{i2}$[min]	26,76	20,13	26,77	13,05	124,35	28,51

In this example can be seen that the structure of the health center is not sufficient. The mean response time of patients in a surgical out-patient clinic is too long (more than 124 minutes). The doctors must quickly carry out examinations and work as hard as they can so as not to obstruct the operation of the whole clinic. The clinic operation can be improved by employing an additional surgeon (it is surprising that mean response time is about 10 minutes).

Example 2

As an example of a closed queueing network we consider a chemotherapy unit of an oncological hospital. Chemotherapy is one of the most common treatments for cancer. The type of chemotherapy treatment depends on many factors, first of all on the type and location of the disease. There are different types of chemotherapy treatment, for instance:
- chemotherapy as the only treatment,
- chemotherapy used in addition to radiation therapy,
- chemotherapy used in addition to surgery (this is known as *neo-adjuvant therapy* or *adjuvant therapy*).

Anticancer drugs destroy cancer cells but healthy cells can be harmed too (side effects). After chemotherapy the healthy cells usually repair themselves.
Usually chemotherapy drugs are given to patient in a day unit at the hospital (having chemotherapy from half an hour to a few hours) but sometimes chemotherapy means a few days' stay at the hospital. Doctors plan the treatment considering several factors. Chemotherapy is usually given as several sessions of treatment. Each cycle of chemotherapy consists of the treatment and the rest period of a few weeks. The number of cycles depends on how cancer responds to the chemotherapy. Sometimes the treatment plan needs to be changed.

We introduce the following classes of patients:
- class „1" – 250 patients of the chemotherapy day unit (having chemotherapy once during one cycle);
- class „2" – 144 patients of a few days' chemotherapy;
- class „3" – 29 additional patients of a few days' chemotherapy;
- class „4" – 50 additional patients of the chemotherapy day unit (having chemotherapy once during one cycle);
- class „5" – 20 patients of the chemotherapy day unit (having chemotherapy five times during one cycle);
- class „6" – 4 additional patients of the chemotherapy day unit (having chemotherapy five times during one cycle)

Our model consists of eight units:
- S_1 – nurse's office: one nurse, exponentially distributed service time with service rate $\mu_1 = \mu_{11} = \mu_{12} = \mu_{15} = \mu_{13} = \mu_{14} = \mu_{16} = 67$;
- S_2 – waiting room: exponentially distributed service time with service rate $\mu_2 = \mu_{21} = \mu_{22} = \mu_{25} = \mu_{23} = \mu_{24} = \mu_{26} = 8$;
- S_3 – doctor's office of a few days' chemotherapy patients: one doctor, exponentially distributed service time with service rate $\mu_3 = \mu_{32} = \mu_{33} = 60$;
- S_4 – doctor's office of a daily chemotherapy patients: four doctors, exponentially distributed service time with service rate $\mu_4 = \mu_{41} = \mu_{45} = \mu_{44} = \mu_{46} = 8{,}33$;
- S_5 – sick-room: two nurses, exponentially distributed service time with service rate $\mu_5 = \mu_{52} = \mu_{53} = 12$;
- S_6 – a few days' chemotherapy unit: 66 beds, exponentially distributed service time with service rate $\mu_6 = \mu_{62} = \mu_{63} = 0{,}218$;
- S_7 – the chemotherapy day unit: 30 beds, exponentially distributed service time with service rate $\mu_7 = \mu_{71} = \mu_{74} = \mu_{75} = \mu_{76} = 1$.

To model the whole cycle of chemotherapy we introduce additional node called home-waiting room S_8 consisting of infinite beds-servers and exponentially distributed service time with service rate different for the classes of patients:

$$\mu_{81} = \mu_{84} = 0{,}092,\ \mu_{82} = \mu_{83} = 0{,}137,\ \mu_{85} = \mu_{86} = 0{,}053.$$

Classes 3, 4, 6 denote classes with the lower priorities at nodes S_6 and S_7.
The routing matrix of chemotherapy unit is shown in Table 3. With the help of Eq. (3) the visit ratios are given as follows:

- $e_{11} = e_{12} = e_{15} = e_{13} = e_{14} = e_{16} = 1$;
- $e_{21} = e_{22} = e_{25} = e_{23} = e_{24} = e_{26} = 1$;
- $e_{32} = e_{33} = 1$;
- $e_{41} = e_{45} = e_{44} = e_{46} = 1$;
- $e_{52} = e_{53} = 1$;
- $e_{62} = e_{63} = 1$;
- $e_{71} = e_{74} = 1$; $e_{75} = e_{76} = 5$;
- $e_{81} = e_{82} = e_{83} = e_{84} = e_{85} = e_{86} = 1$.

We compute the throughput and other performance measures using the summation method (nodes 1, 2, 3, 4, 5, 8) and the extended summation method (nodes 6, 7). To compute throughput λ_r we use the method described in previous section. Using Eq. (9) and (11) we finally get:

$$\lambda_1 = 20{,}2833;\ \lambda_2 = 11{,}6342;\ \lambda_3 = 1{,}9021;$$
$$\lambda_4 = 3{,}4889;\ \lambda_5 = 0{,}8064;\ \lambda_6 = 0{,}1148.$$

The utilizations of each node can be computed with Eq. (1) and (7):

$$\rho_1 = 0{,}5706;\ \rho_2 = 4{,}7787;\ \rho_3 = 0{,}2256;\ \rho_4 = 0{,}7411;$$
$$\rho_5 = 0{,}5640;\ \rho_6 = 0{,}9408;\ \rho_7 = 0{,}9460;\ \rho_8 = 374{,}5818.$$

Using Little's theorem and Eq. (8), (10) and (11) the mean waiting times of patients of rth class at the ith node (W_{ir} = the mean response time – service time) are described in Table 4.

We notice that the mean waiting times of priority classes are too long at the nodes S_6 (more than 3 days) and S_7 (more than 2 days).

Table 3: The Routing Matrix

i, r	1,1	1,2	1,3	1,4	1,5	1,6	2,1	2,2	2,3	2,4	2,5	2,6	3,2	3,3	4,1	4,4	4,5	4,6	5,2	5,3	6,2	6,3	7,1	7,4	7,5	7,6	8,1	8,2	8,3	8,4	8,5	8,6
1,1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1,2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1,3	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1,4	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1,5	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1,6	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2,1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2,2	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2,3	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2,4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2,5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2,6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3,2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3,3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
4,1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
4,4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
4,5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
4,6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
5,2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
5,3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
6,2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
6,3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
7,1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
7,4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
7,5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,8	0	0	0	0	0	0,2	0
7,6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0,8	0	0	0	0	0	0,2
8,1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8,2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8,3	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8,4	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8,5	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8,6	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 4: The Mean Waiting Times

W_{ir}	Class 1	Class 2	Class 3	Class 4	Class 5	Class 6
S_1	0,0197	0,0197	0,0197	0,0197	0,0197	0,0197
S_2	0	0	0	0	0	0
S_3	0	0,0048	0,0048	0	0	0
S_4	0,0570	0	0	0,0570	0,0570	0,0570
S_5	0	0,0388	0,0388	0	0	0
S_6	0	0,1876	3,0565	0	0	0
S_7	0,1191	0	0	2,1247	0,1191	2,1247
S_8	0	0	0	0	0	0

SUMMARY

Queueing networks can find wide application in the health care area. The queueing network analysis showed the bottleneck of considered clinics. We obtained interesting performance measures such as patients' mean waiting time and the mean number of patients. These results may help other researches, practitioners and managers obtain good results of the modeling of similar structures.

REFERENCES

Albin S.L.; J. Barrett; D. Ito; and J.E. Mueller. 1990. "A queueing network analysis of a health center". *Queueing Systems* 7, 51-61.

Baskett F.; K. Chandy; R. Muntz; and F. Palacios. 1975. "Open, closed and mixed networks of queues with different classes of customers". *Journal of the ACM* 22, No.2, 248-260.

Bolch G.; S. Greiner; H. de Meer; and K.S.Trivedi. 1998. *Queueing networks and Markov chains. Modeling and performance evaluation with computer science applications.* John Wiley&Sons, INC.

Buzen J.P. 1973."Computational algorithms for closed queueing networks with exponential servers". *Communications of the ACM* 16, No.9., 527-531.

Czachórski T. 1999. *Modele kolejkowe w ocenie efektywności sieci i systemów komputerowych.* Pracownia Komputerowa Jacka Skalmierskiego, Gliwice.

Doi M; Y.M. Chen; and H. Ōsawa. 1997. "A queueing model in which arrival times are scheduled". *Operations Research Letters* 21, 249-252.

Filipowicz B. 1997. *Modelowanie i analiza sieci kolejkowych.* Wydawnictwa AGH, Kraków.

Kleinrock L. 1976.*Queueing systems, volume 2, Computer application.* John Wiley&Sons, New York.

Kwiecień J. 2004. *Zastosowanie sieci kolejkowych z wieloma klasami zgłoszeń w zagadnieniach organizacyjnych służby zdrowia*, Ph.D. dissertation, AGH-UST, Kraków, not published.

Reiser M. and S. Lavenberg. 1980. "Mean value analysis of closed multichain queueing networks". *Journal of the ACM* 27, No.2, 313-322.

EVALUATION OF FLOWS IN COMMUNICATION NETWORKS WITH DIFFERENT MECHANISMS FOR SUPPORT OF QUALITY OF SERVICE

Daniela Traczinski, Wolfram Luther, Nicola Ehling
Institut für Informatik und Interaktive Systeme
Universität Duisburg-Essen
D 47048 Duisburg
{d.traczinski, luther}@informatik.uni-duisburg.de
nicola.ehling@uni-duisburg.de

Gerhard Haßlinger
Deutsche Telekom, T-Systems
Am Kavalleriesand 3
D 64307 Darmstadt
gerhard.hasslinger@telekom.de

KEYWORDS

Class based scheduling in routing and switching, prioritization, fair share of bandwidth, traffic rate distribution and variability, data loss.

ABSTRACT

In this paper we give an outline of an analytical modeling approach for routers and switching systems which implements QoS support mechanisms including verified workload computation and realistic scheduling and reservation strategies. Traffic flows are represented by the distribution of the quantity of traffic arriving per time slot. For a number of concurrent flows being transmitted together over channels of limited capacity we study the impact of (un)limited buffers on the flows which experience differentiated services.

INTRODUCTION

Analysis methods for multiplexing, routing and switching systems integrate the input and service process into a corresponding Markovian state model. Implications on QoS parameters like delay, data loss and buffer occupancy then become apparent from the steady state solution of the system. While the QoS analysis is simple to handle without buffers, it becomes complex and requires elaborate queueing theoretic approaches with regard to computational expense and stability when buffers are included. Depending on the model, there are different ways to determine workload distributions. Simple GI/GI/1 systems or Semi-Markov Processes (SMP) can be used to model the distribution function of arriving data as well as the autocorrelation function. The analysis of such models and the modeling itself should be done as exactly as possible, since the results have influence on service level agreements.

Matrix-analytic approaches have been established by M.F. Neuts (Neuts 1989) as a standard method and have been improved in many ways in the last decades. A second class of methods may be denoted as factorization approaches. The difference between both principles is that factorization approaches often have a lower computational complexity, i.e. a quadratic increase in computation time and a modest increase in storage occupancy with regard to the number of stages combined with poor convergence properties, whereas a cubic computation time and a higher storage consumption is required by comparable matrix analytic approaches together which assured properties for numerical stability in computation.

There are two main factorization approaches, the direct factorization by determination of h roots of the characteristic polynomial of degree $g+h$, where g and h are the number of phases or stages of the arrival and the service time distribution, and the Wiener Hopf method which leads to an equivalent form of the steady state solution represented on the basis of generating functions. Both approaches have some drawbacks concerning numerical stability in a straightforward implementation: The computation of the characteristic polynomial's roots normally accumulates errors, so that for larger degree of the polynomial the computed roots become more and more inaccurate.

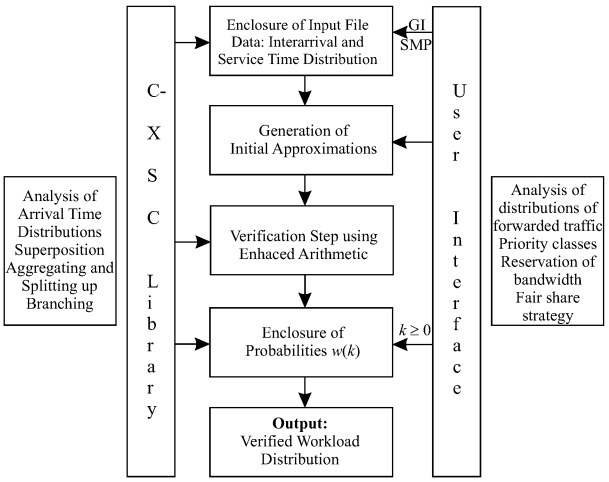


Figure 1: Interval based verified numerical analysis of the traffic flow distribution in multi service communication networks (InterVerdiCom)

Even the decision, which of the computed roots are located inside the unit disk and therefore are relevant for the solution may become uncertain. On the other hand, the efficient algorithm of Grassmann and Jain is not proven to be convergent in the SMP/GI/1 case. Therefore we have implemented both factorization approaches for the GI/GI/1 and the SMP/GI/1 case with regard to achieve verified results by using interval arithmetic (Fausten et al. 2004a,c). The implementation of Wiener-Hopf factorization for systems with semi-Markovian arrival with result verification is described in a recent paper (Fausten and Haßlinger 2004). Wiener-Hopf factorization is an efficient alternative with low computational complexity to analyze especially the workload and waiting time distribution of a buffered switching system.

The computation of a verified workload distribution uses C-XSC and the associated toolbox (C-XSC Download 2004). C-XSC is a C++ class library providing interval arithmetic. The toolbox consists of instruments to find verified solutions for basic numerical problems. Both methods lead to verified enclosures for workload distributions even in the case of large state spaces. The main advantage of these approaches is the guaranteed accuracy of the results, which is important for quality of service evaluation.

The adaptation of discrete time Markovian state models for performance evaluation of multiplexed video streams shows that an accurate match of the distribution and short term correlation is sufficient for a reasonable prediction of data loss and delay characteristics for real time constraints for few seconds. Including long term correlation should improve the modeling on account of larger model state space. Thus, splitting up theses processes in smaller ones or approximating them by processes with smaller state spaces is actually under examination.

To complete our system, we include tools which implement an analytic approach for time varying traffic flows through a network by calculating the distributions of the traffic rate and the corresponding loss rate due to overload at switching systems and routers including the Intserv and Diffserv QoS support principles (Fausten et al. 2004b). The Intserv approach requires signalling and state information to be maintained in each router on the path in order to provide resource reservation per flow (Wroclawski 1997). Differentiated services represent a simpler concept based on classification and prioritization of traffic (Carpenter and Nichols 2002). Each IP packet is marked to belong to a specific class using a 6 bit field in the header such that 64 classes can be distinguished. The classification is done by edge routers, whereupon core routers of a Diffserv network can set up class based scheduling. Main scheduling principles are strict priorities, class based reservation and a fair scheduling traffic for flows within the same class. Then the distribution of the discrete incoming traffic of a number of sources $Q_i, i = 1, \ldots, N$, can be used to compute the distribution of a discrete random variable for the quantity of traffic passing a common bottleneck and arriving at the edge K_i in a time slot depending on the assumed priority policy.

So we derived the transformation in the distribution of the quantity of traffic flows, when traffic is forwarded by a switching system subject to data loss in overload without buffering. Reservation, prioritization and a fair bandwidth sharing have been included as alternatives for support of QoS. In this paper (un)limited buffers are used which are emptied by forwarding the packets stored for one or more time steps depending on the priority policy or by dropping them. The analysis of data loss and the output process of a switching system is the main part of a network-wide analysis of routing with differentiated treatment of traffic flows, which is presently being implemented in our system InterVerdiCom. Evaluation of the analytical model gives a better insight in the interrelation of traffic flows and on the impact of other flows and their variability on the data loss rate of a considered flow.

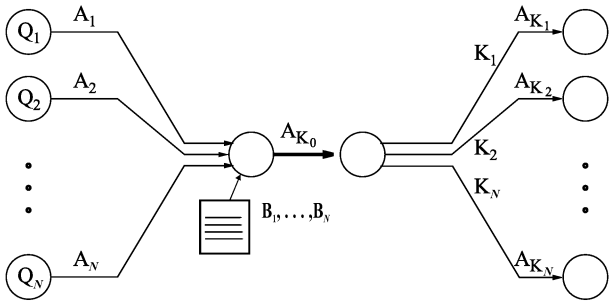


Figure 2: Modeling traffic flows over a common bottleneck link

BASIC TRAFFIC FLOW MODEL

The main steps of analysis in a decomposition approach determine the aggregation of a number of traffic flows entering a switching system, the evaluation of the switch performance with regard to waiting times and data loss when forwarding the traffic flow, and the branching of traffic into subflows with different destinations departing from a switch, which appear as the input at other network nodes (Haßlinger and Rieger 1996). In this environment the traffic on each network link is represented in an appropriate form by aggregation and branching starting from the source traffic.

If an edge K_0 with a limited bandwidth c_{max} forwards the arrivals of sources $Q_1, ..., Q_N$, then simultaneously arriving packets may provoke a temporary overload leading to data loss. By applying a selection following a well defined strategy the amount of data is reduced to meet the capacity of the edge or to be buffered for an appropriate time segment.

In the following, we will study different selection strategies for dropping packets coming from the sources Q_i and the impact they cause when the aggregated flow is again separated into the corresponding subflows. We denote by

- A_i a discrete random variable for the quantity of traffic, i.e. the number of data units sent by source Q_i in a time slot,
- B_i a discrete random variable for the quantity of traffic buffered for source Q_i,
- $\tilde{A}_i$ a discrete random variable for the quantity of traffic resulting from the number of packets sent by source Q_i and its number of buffered packets. $\tilde{A}_i$ describes a transformation of the source flow A_i by the buffer.

We assume the traffic quantity of the sources to be bounded by M, which may correspond to a maximum access line speed and set $P(A_i = s) = 0$ for $s > M$. The computational complexity of the analysis mainly depends on M.

Traffic sources are assumed to be independent. Then the distribution function for the buffer contents of a regarded source is computed using the distributions of the packets sent by the sources.

UNLIMITED BUFFERS

Our buffer should store the data packets for just one unit of time, data that does not get transmitted afterwards will be completely lost. The term $P(A_i^t = k)$ responds to the probability that the sum of packets sent by source Q_i and the number of buffered packets at time t of this source equals k.

In the following we denote by $P(B_i^t = k)$ the probability of an amount of k packets buffered for source Q_i at time t. We compute $P(\tilde{A}_i^t = k)$ as convolution:

$$P(\tilde{A}_i^t = k) = P(A_i^t + B_i^{t-1} = k) \; .$$

Then for the quantity A_{K_i} of traffic arriving at the edge K_i in a time slot the formulas developed in (Fausten et al. 2004b) are valid with the minor change that instead of A_i the variable $\tilde{A}_i$ should be used.

Source traffic with strict priorities

This strategy forwards traffic with lower priority only if packets sent by sources with higher priority do not exceed the edge's capacity. Then the remaining bandwidth can be used, excessive traffic is lost.

We refer to figure 2 with decreasing order of priorities from highest for source Q_1 to lowest priority for Q_N. Then the discrete distribution function of the buffer content for source Q_i results from the corresponding source traffic A_i with regard to preferred traffic from sources $Q_1, \ldots, Q_{i-1}$:

$$P(B_i^{t-1} = 0) = P(A_i^{t-1} = 0) + \sum_{m=1}^{c_{max}} P(A_i^{t-1} = m) P(\sum_{j<i} A_j^{t-1} \leq c_{max} - m)$$

$$P(B_i^{t-1} = k) = \sum_{m=0}^{c_{max}} P(A_i^{t-1} = m + k) P(\sum_{j<i} A_j^{t-1} = c_{max} - m)$$

for $k > 0$.

Fair share by random packet drop

Another strategy for dropping traffic in overload is to randomly select $x - c_{max}$ packets out of x arriving packets ($x > c_{max}$) such that each packet has the same chance to be dropped for any of the involved flows.

To calculate the distribution of each buffer part B_i we determine the probabilities $P(B_i = k)$, $1 \leq k \leq M$, by using the hypergeometric distribution in the case of overload. This provides the number of possibilities to select m out of c_{max} packets being forwarded for source A_i and another $c_{max} - m$ packets coming from the remaining sources A_j, $j \neq i$:

$$P(B_i^{t-1} = 0) = P(A_i^{t-1} = 0) + \sum_{m=1}^{c_{max}} P(A_i^{t-1} = m) P(\sum_{j \neq i} A_j^{t-1} < c_{max} - m) + \sum_{m=1}^{c_{max}} P(A_i^{t-1} = m) \sum_{\substack{n= \\ c_{max}-m}}^{(N-1)M} P(\sum_{j \neq i} A_j^{t-1} = n) \cdot \frac{\binom{m}{m}\binom{n}{c_{max}-m}}{\binom{m+n}{c_{max}}}$$

$$P(B_i^{t-1} = k) = \sum_{m=0}^{c_{max}} P(A_i^{t-1} = m + k) \cdot \sum_{\substack{n= \\ c_{max}-m}}^{(N-1)M} P(\sum_{j \neq i} A_j^{t-1} = n) \frac{\binom{k+m}{m}\binom{n}{c_{max}-m}}{\binom{k+m+n}{c_{max}}}$$

for $k > 0$.

BUFFER WITH LIMITED CAPACITY

In the following, we denote by b_{max} the maximum capacity of the buffer, i. e. the number of packets from all sources that can be stored. The share of buffer capacity for an individual source depends on the traffic amount of the remaining sources and their amount of packets getting stored in case of overload. Therefore the results vary for the different strategies:

Source traffic with strict priorities

By allocating the buffer capacity responding to the priorities of the sources, $P(B_i = k)$ refers to the probability of the following cases: Q_i uses the remaining capacity of the transmission link, further k packets sent by Q_i are stored; or the capacity c_{max} is completely used by sources with higher priority and the regarded source Q_i sends exactly k packets all getting stored, or the traffic amount of source Q_i exceeds k packets but gets lost because the privileged sources are using the buffer capacity up to k packets.

$$P(B_i^{t-1} = 0) = \sum_{m=0}^{c_{max}} P(A_i^{t-1} = m) P(\sum_{j<i} A_j^{t-1} \leq c_{max} - m) + \sum_{m=1}^{M} P(A_i^{t-1} = m) P(\sum_{j<i} A_j^{t-1} \geq c_{max} + b_{max})$$

$$P(B_i^{t-1} = k) = \sum_{m=1}^{c_{max}} P(A_i^{t-1} = k + m) P(\sum_{j<i} A_j^{t-1} = c_{max} - m) + P(A_i^{t-1} = k) \sum_{n=0}^{b_{max}-k} P(\sum_{j<i} A_j^{t-1} = c_{max} + n) + \sum_{l=1}^{M} P(A_i^{t-1} = k + l) P(\sum_{j<i} A_j^{t-1} = c_{max} + b_{max} - k)$$

for $k > 0$.

Fair share by random packet drop

Since all sources are treated equally, it is possible that in case of overload one fraction of packets sent by the regarded source is chosen for transmission, another fraction gets some amount of the available buffer capacity and the remaining packets are dropped. In the following we denote by m the number of packets of the regarded source that are chosen for transmission, and with $\tilde{m}$ the number of packets that are dropped due to no available buffer capacity.

$$P(B_i^{t-1} = 0) = P(A_i^{t-1} = 0) + \sum_{m=1}^{c_{max}} P(A_i^{t-1} = m) \cdot P(\sum_{j \neq i} A_j^{t-1} < c_{max} - m) + \sum_{m=1}^{c_{max}} P(A_i^{t-1} = m) \cdot \sum_{\substack{n= \\ c_{max}-m}}^{(N-1)M} P(\sum_{j \neq i} A_j^{t-1} = n) \frac{\binom{m}{m}\binom{n}{c_{max}-m}}{\binom{m+n}{c_{max}}} + \sum_{m=0}^{c_{max}} \sum_{l=1}^{M} P(A_i^{t-1} = m + l) \sum_{\substack{n=c_{max} \\ -m+b_{max}}}^{(N-1)M} P(\sum_{j \neq i} A_j^{t-1} = n) \cdot \frac{\binom{m+l}{m}\binom{n}{c_{max}-m}}{\binom{m+l+n}{c_{max}}} \cdot \frac{\binom{l}{0}\binom{n-c_{max}+m}{b_{max}}}{\binom{l+n-c_{max}+m}{b_{max}}}$$

$$P(B_i^{t-1} = k) = \sum_{m=0}^{c_{max}} P(A_i^{t-1} = m + k)$$

$$\cdot \sum_{\substack{n= \\ c_{max}-m}}^{(N-1)M} P(\sum_{j\neq i} A_j^{t-1} = n) \frac{\binom{k+m}{m}\binom{n}{c_{max}-m}}{\binom{k+m+n}{c_{max}}}$$

$$\cdot \frac{\binom{k}{k}\binom{n-c_{max}+m}{b_{max}-k}}{\binom{k+n-c_{max}+m}{b_{max}}} + \sum_{m=0}^{c_{max}} \sum_{l=1}^{M} P(A_i^{t-1} = m+l+k)$$

$$\cdot \sum_{\substack{n=c_{max}-m \\ +b_{max}-k}}^{(N-1)M} P(\sum_{j\neq i} A_j^{t-1} = n) \frac{\binom{k+m+l}{m}\binom{n}{c_{max}-m}}{\binom{k+m+l+n}{c_{max}}}$$

$$\cdot \frac{\binom{k+l}{k}\binom{n-c_{max}+m}{b_{max}-k}}{\binom{k+l+n-c_{max}+m}{b_{max}}} \text{ for } k > 0.$$

CONCLUSION

We have summarized some important results of our current work. Furthermore, we derived the transformation in the distribution of the quantity of traffic flows, when traffic is forwarded by a switching system subject to data loss in overload with (un)limited buffers. Prioritization and a fair bandwidth sharing have been included as alternatives for support of QoS. The analysis of buffers and the output process of a switching system is the main part of a network-wide analysis of routing with differentiated treatment of traffic flows, which is presently being implemented. Evaluation of the analytical model can give insight in the interrelation of traffic flows and on the impact of other flows and their variability on the data loss rate of a considered flow. For future work remains the expansion to storing the buffered packets for more than one time slot using the recursive relation $B_i^{t-2} = B_i^{t-1} - A_{K_i}$.

ACKNOWLEDGEMENTS

This research and software development for algorithms with result verification has been carried out by the authors in a recent project VERDIKOM funded by the German Research Council (DFG).

REFERENCES

Carpenter, B. E. and Nichols, K. 2002. "Differentiated Services in the Internet" Proceedings of the IEEE (2002) 1479-1494.

CXSC. 2004. http://www.math.uni-wuppertal.de/wrswt/xsc-sprachen.html .

Fausten, D., Luther, W. and Haßlinger, G. 2004. "Accurate computation of traffic workload distributions", accepted and in print for Numerical Algorithms, Selected papers of SCAN 2002, Kluwer Akad. Publ.

Fausten, D., Ehling, N., Luther, W. and Haßlinger, G. 2004. "Modeling and analysis of switching systems in multi-service networks with QoS support" 18th European Simulation Multi-Conference (ESM), Magdeburg (June 2004) 160-165.

Fausten, D., Luther, W. and Haßlinger, G. 2004. "Computing the workload distribution of discrete time semi-Markov servers with result verification" 12th GI/ITG Conference on Measuring, Modelling and Evaluation of Computer and Communication Systems (MMB/PGTS) Dresden (2004) 245-254.

Fausten, D. and Haßlinger, G. 2004. "Verified Numerical Analysis of the Performance of Switching Systems in Telecommunication" in Numerical Software with Result Verification, LNCS 2991, Springer (2004) 206-225.

Haßlinger, G. and Rieger, E. 1996. "Analysis of open discrete time queueing networks: A refined decomposition approach" Journal of the Operations Research Society 47 (1996) 640-653.

Neuts, M. F. 1989. "Structured Stochastic Matrices of M/G/1 Type and their Applications" M. Dekker.

Wroclawski, J. 1997. "The Use of RSVP with IETF Integrated Services" proposed IETF standard, RFC 2210, http://www.rfc-editor.org/cgi-bin/rfcsearch.pl (search for RFC 2210).

AUTHOR BIOGRAPHIES

N. EHLING studied Mathematics at the University of Duisburg-Essen. Her diploma thesis dealt with the analysis of traffic models, especially semi-Markov processes.

G. HASSLINGER received the diploma and doctoral degree from Darmstadt University of Technology, during his work as a researcher at the computer science department. In 1996 he joint Deutsche Telekom, where he is involved in projects for the future design of the Internet platform. His research interests include performance, reliability and QoS aspects of communication networks, queueing network modeling and analysis, MPLS traffic engineering and error detecting and correcting codes.

W. LUTHER received a degree of doctor in 1973 at the University of Marburg, and wrote his habilitation thesis at the Technical University of Aachen in 1980. He is now professor for Computer Science at the University of Duisburg-Essen. His research fields are scientific computing, computer graphics and image processing as well as interactive multimedia systems. Actually he carries out several R&D projects funded by the DFG.

D. TRACZINSKI. Since graduating in Mathematics at the University of Duisburg-Essen, Daniela Traczinski is employed there at the department of computer science. Actually she works at the project VERDIKOM on traffic modeling in multi service networks and the analysis of resource requirements for quality of service. Her main research interest is scientific computation.

PERFORMANCE EVALUATION OF SOFTWARE ARCHITECTURES WITH QUEUING NETWORK MODELS

Simonetta Balsamo Roberto Mamprin Moreno Marzolla

Dipartimento di Informatica, Università Ca' Foscari di Venezia
via Torino 155, 30172 Mestre (VE), Italy
E-mail: balsamo@dsi.unive.it rmamprin@dsi.unive.it marzolla@dsi.unive.it

KEYWORDS

Unified Modeling Language, Queuing Networks, Software Performance Evaluation, Distributed Systems

ABSTRACT

We propose an approach for software performance modeling based on UML software specifications and queuing network performance models. We consider the integration of performance and specification model to provide a tool for quantitative evaluation of a software architecture at the design phase in the software development cycle. The approach derives a performance model starting from an annotated Unified Modeling Language (UML) specification, based on a subset of the standard UML Profile for Schedulability, Performance and Time Specification. More specifically, we consider a set of UML diagrams, i.e., Use Case, Activity and Deployment diagrams, and we propose an algorithm for deriving a product-form queuing network performance model. Then the queuing network model is easily analyzed with product-form algorithms to obtain a set of performance indices that are used to provide feedback at the software architectural design level. The analysis cycle can be iterated to meet given performance goals or to compare different software alternatives. The approach has been implemented as a prototype tool written in Java.

1 INTRODUCTION

Large software systems play a vital role in many commercial and industrial infrastructures. Complex e-commerce sites, data acquisition networks and distributed computing environment are very expensive to develop and maintain. Such systems should provide an adequate level of performances in order to be used. Early performance analysis can help to identify and correct problems from the early stages of the software development life cycle, in order to compare design alternatives or to identify system bottlenecks. Early identification of performance problems is desirable as the cost of design change increases with the later phases in the software development cycle.

Performance evaluation should be integrated in the software development process (Balsamo et al., 2004b; Smith, 1990; Smith and Williams, 2002). The SPE approach by Smith (Smith, 1990) was the first integrated approach for development and performance evaluation of software systems. Several approaches have been proposed in the last years that can be classified by considering various characteristics, including the software and the performance model. A comprehensive review of the software performance methods can be found in (Balsamo et al., 2004b).

Many approaches consider UML (Object Management Group (OMG), 2001) as the software system specification. This choice is motivated by the fact that UML has a large users base, and is able to model many different aspects (static, dynamic, behavioral) of a system. Moreover, UML provides standard extension mechanisms based on additional constructs. Recently the UML Profile for Schedulability, Performance and Time Specification has been adopted as an OMG standard (Object Management Group (OMG), 2002a). It allows the definition of requirements for performance and scheduling analysis, and the specification of quantitative informations directly in the UML model. Some recent approaches for software performance analysis are based on the UML Profile (Xu et al., 2003). As the performance model is concerned, various methods consider formal and analytical models such as Queuing Network (QN), stochastic process algebras or stochastic Petri nets, and some are based on simulation. It has been observed that QN are the preferred model, taking advantage of the high level of abstraction of the QN formalism that allows easier mapping and feedback of software systems, especially in computer-based software development process (Balsamo et al., 2004b; Grassi and Mirandola, 2004).

In this paper we propose an approach for software performance modeling based on UML as the software description notation, and Queuing Networks (Kleinrock, 1975) as the performance model. The motivation of the choice of QN models is twofold. First, the high level of abstraction of the model makes it easy to define a direct correspondence between software components and performance model elements, so that we can easily derive the QN from UML diagram specification. This allows us to provide a direct feedback of performance results at the software design level. The second motivation is that we aim to identify a simple product-form QN (Kleinrock, 1975) in order to obtain performance measures by computationally efficient algorithms (Reiser and Lavenberg, 1980).

We consider the software performance evaluation cycle

described by the following steps:

1. Definition of the performance requirements of the software system.
2. UML specification of the software system.
3. Transformation of the software specification into a performance model based on QN, using a suitable transformation algorithm.
4. Analysis of the QN and derivation of performance indices.
5. Feedback of performance results on the software model elements.
6. Analysis of the software performance results and possible iteration from step 2 (e.g., if the performance requirements are not met or different architectural implementations have to be examined)

Starting from the UML software model annotated with performance-oriented annotations based on the *UML Performance Profile* (Object Management Group (OMG), 2002a), we consider UML Use Case, Activity and Deployment diagrams. We define an algorithm for translating the annotated software model into a QN based performance model. The performance model is solved using an appropriate solution technique and performance results provide feedback at the software specification level by the annotations. Hence the software performance analysis cycle can be iterated to meet given performance requirements or to compare different software design alternatives.

The proposed approach can be integrated with other different methods for software performance analysis in a more general environment that can provide a set of tools for quantitative analysis of software systems. Specifically, as observed in (Balsamo et al., 2004a) software designers would take advantage of the combined use of different methodologies to evaluate the performance of software artifacts. To this aim, for example, the proposed approach can be easily integrated with the recently developed tool UML-Ψ (Balsamo et al., 2004a) providing a simulation-based evaluation of UML-based software systems.

The proposed method has been implemented as s Java program called UML Queuing Network Evaluator (UML-QNE). UML-QNE parses the annotated model produced by a UML CASE tool. In particular, we use the freely available ArgoUML tool (ArgoUML), that can export the software model in XML Metadata Interchange (XMI) format, which is a standard XML-based representation of UML models. UML-QNE parses the XMI file, builds the QN model and applies the MVA algorithm (Reiser and Lavenberg, 1980). Fig. 1 illustrates the steps described above and the structure of the proposed approach. In particular, steps 3 through 6 correspond to the UML-QNE Java tool.

The paper is organized as follows. In Section 2 we briefly discuss the proposed approach with respect to some recent model-based methods for software performance analysis. Section 3 presents the proposed approach by introducing the software model based on the UML Profile, the QN

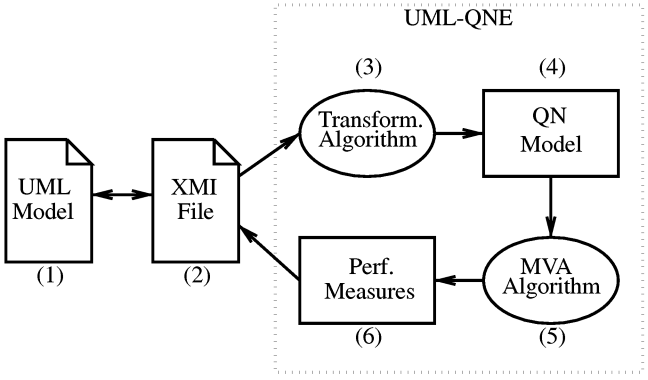


Figure 1: Software Performance Steps and UML-QNE Structure

performance model and the proposed translation algorithm. The method implementation is described in Section 4 and a simple case study are described in Section 5. Finally, Section 6 presents conclusions and open problems.

2 SOFTWARE PERFORMANCE EVALUATION

Different approaches dealing with performance evaluation of software systems have been proposed in the literature (Proceedings of WOSP 2000; Proceedings of WOSP 2002; Proceedings of WOSP 2004). In the following, we briefly recall some approaches based on UML as the software specification notation.

King and Pooley in (King and Pooley, 2000) derive performance models based on Generalized Stochastic Petri Nets from UML collaboration and statechart diagrams. In (Pooley and King, 1999), Pooley and King describe how the various kinds of UML diagrams can be used for performance evaluation purposes. The approach adds textual notations to UML diagrams to include useful information for performance evaluation (e.g., time labels in sequence diagrams). Such annotations are used to produce more complete models of software systems. Bernardi et al. (Bernardi et al., 2002) derive a SPN model from UML state and sequence diagrams. Cortellessa and Mirandola present in (Cortellessa and Mirandola, 2002) a methodology for translating UML sequence, use case and Deployment diagrams into performance model based on Extended Queuing Networks, using an intermediate transformation into Execution Graphs (Smith, 1990). Gomaa and Menascé (Gomaa and Menascé, 2001) derive a QN-based performance model from UML class and collaboration diagrams to represent the interconnection pattern of a distributed software architecture. Gu and Petriu (Gu and Petriu, 2002) and Petriu and Shen (Petriu and Shen, 2002) derive performance models based on Layered Queuing Network models from a description of Software Architecture (SA) based on annotated UML Activity diagrams. Lindemann et al. (Lindemann et al., 2002) develop an algorithm for deriving performance models based on Generalized Semi-Markov Processes from UML State and Activity

diagrams. Kähkipuro (Kähkipuro, 2001) proposes a framework on UML notation for describing performance models of component-based distributed systems. The performance model is based on Augmented Queuing Networks.

A few works also consider simulation-based performance models for software systems (Arief and Speirs, 2000; De Miguel et al., 2000; Hillston and Wang, 2003; Marzolla, 2004). In general, simulation-based approaches derive a simulation model from UML specifications; the simulation model is implemented as a simulation program which is then executed. Simulation results are computed as confidence intervals for some figures of merit (e.g., utilization and throughput of resources, average execution times).

Analytical performance models have some advantages over those based on simulation. Some classes of analytical models can be solved very efficiently either exactly or by approximation. Moreover, some analytical models allow parametric solution, with respect to some parameter of the performance mode. Hence, in this case we can easily carry on parametric performance analysis under different model assumptions (e.g., by varying parameters' value), avoiding the re-evaluation of the entire model. On the other side, generally speaking, system performance analysis with simulation-based models leads to a high computation time cost for model development, validation and execution. Moreover, simulation provides a solution in terms of estimated values, with confidence intervals. However, the main advantage of simulation models is their generality and wide applicability, i.e., any system which can be described can also be analyzed by simulation, while analytical models require specific assumptions, and efficient solution algorithms apply under special constraints (Lavenberg, 1983). Efficient solution algorithms are only available for analytical models satisfying certain constraints.

We propose an approach that derives a QN model from annotated UML specifications. The software specification model must satisfy relatively mild constraints (service demands can only be exponentially distributed). Under these assumptions, the QN model is in product-form and can be efficiently solved by using the MVA algorithm (Reiser and Lavenberg, 1980). Differently from other approaches, we consider the direct derivation of a simple product-form network from a UML software specification model based on a variation of the standard UML Performance Profile (Object Management Group (OMG), 2002a).

3 THE PROPOSED APPROACH

In this section we present the proposed approach by introducing the annotated UML software specification model, the QN performance model and the transformation algorithm (step 3 in Fig. 1).

3.1 The software model

The software system is described in term of the following UML diagrams: Use Case diagrams, representing workloads applied to the system; Deployment diagrams, describing the available physical resources (processors) where computations take place; Activity diagrams, describing both the order in which resources are used, and the corresponding service demand.

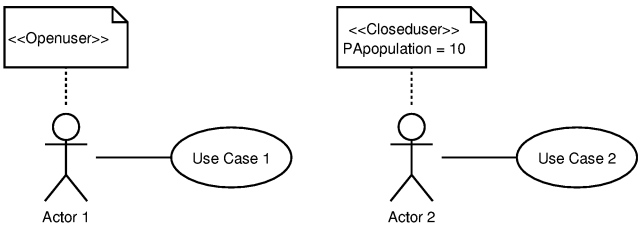


Figure 2: Example of annotated Use Case diagrams

Use case diagrams Each actor in a Use Case diagram may represent a stream of requests arriving at the system. There may be an unlimited sequence of requests (open workload), or a fixed population of users requiring service from the system (closed workload). Actors representing open workloads are stereotyped as $\ll$ OpenUser $\gg$, while actors representing closed workloads are stereotyped as $\ll$ ClosedUser $\gg$. $\ll$OpenUser$\gg$ actors

For the latter class of actors it is necessary to specify the total number of requests circulating in the system; this is done with the `PApopulation` tagged value associated to the actor. Multiple actors may be present in the same system, meaning that there are multiple concurrent streams of requests. Each actor has an associated use case representing the computations triggered by each actor. The details of the computations are described by an Activity diagram which must be associated to each Use Case.

Fig. 2 shows an example of annotated Use Case diagram, where Actor 1 represents an open workload and Actor 2 represents a closed population of 10 requests circulating in the system.

Deployment diagrams Deployment diagrams are used to model the physical resources available in the system. Each resource is represented by a node in the Deployment diagrams. Each node, which must be stereotyped as $\ll$node$\gg$, represents a processor with a given scheduling policy. The following tagged values can be associated to Deployment diagram nodes:

`PAschedpolicy` Defines the scheduling policy of that processor, which can be one of "FIFO" (First-In-First-Out), "LIFO" (Last-In-First-Out) and "PS" (Processor Sharing).

`PArate` defines the processing rate (speed) of the processor. This means that a user with a service request of S time units will complete in $S/PArate$ time units.

`PAservers` defines the number of processors concurrently executing the requests. This tag can be used to represent a multiprocessor resource having N equal proces-

sors executing in parallel. All the processors share the same queue of pending requests.

Fig. 3 shows an example of annotated Deployment diagram. Each node represents a processor with the characteristics described with the tagged values.

Activity diagrams Having described the workloads and the physical resources available in the system, it is now necessary to specify how the resources are used, that is, which computations are performed in the system. Computations are described by associating an Activity diagram to Use Cases. In this way, the workload represented by an actor will trigger the computation represented by the Activity diagram associated to the corresponding Use Case.

Each action state of an Activity diagram, stereotyped as ≪ ServiceCenter ≫, represents a computation which requires service to one resource. The following tagged values can be specified to provide informations for building the performance model:

`PAresource` is the name of the resource (node in one of the Deployment diagrams) from which service is requested.

`PAoccurrence` represents the interarrival time of this service request. The tag can be specified only if the QN performance model is an open QN (described in the following).

Action states are linked each other with a predecessor-successor relationship. This relationship is used to model the sequence of computations which are executed by a request. In general, one action state may have multiple successors. In this case, each transition must be labeled with the probability that the transition is traversed. This models nondeterminism in the sequence of actions which are executed.

Fig. 4 represents an example of annotated Activity diagram. Transitions are annotated with the `PAprob` tag showing the probability of traversing the corresponding arc. Values of the `PAresource` tags refers to node names of the Deployment diagram in Fig. 3.

Note that our approach does not support synchronization bars in Activity diagrams. Synchronization bars are used to split the execution flow in multiple, concurrent threads or to join multiple threads into a single flow. While synchronization bars could be translated into fork and join nodes in the QN model, the resulting QN would not be in product form, thus requiring more complex solution algorithms, which may produce only approximate results instead of exact ones.

3.2 The Performance Model

We derive the performance model from the UML specification, as sketched in Section 1, by defining a multiclass QN (Lavenberg, 1983). A Queuing Network is a set of service centers (also called nodes or service stations). Each service center is formed by a queue and a set of identical servers.

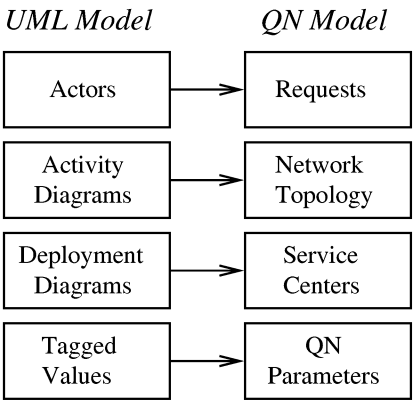


Figure 5: Mapping Between UML and Performance Model Elements

Requests join the queue and are serviced by the first server becoming idle, according to a specific queuing discipline.

UML model components are translated into the corresponding QN model elements as follows. Each node in the Deployment diagrams defines a service center. From Actors in Use Case diagrams we identify the type of QN model, e.g., open, closed or mixed QN. Finally, from Activity diagrams we derive the network topology, that is the behavior of the classes of users circulating through the system. The mapping between UML and performance model elements is illustrated in Fig. 5.

Open QN can be generated from Use Case diagrams in which the actor is stereotyped as ≪ OpenUser ≫. Note that no tagged values are defined for this kind of actor, as externally arriving requests can be defined by the `PAoccurrence` associated to each action state. This means that the action state can be executed also by an external stream of requests reaching the system at the given rate. Closed QN are made of a number of service centers in which a fixed population of requests circulates. Each request receives service from a service center, after which it is routed to a (possibly different) another service center. This kind of network is generated from an actor stereotyped as ≪ ClosedUser ≫, where the associated `PApopulation` tag is used to specify the number of requests. Mixed QN are an extension of the previously described networks: in mixed networks there are R different classes of circulating requests. Each class of requests moves through the network differently from requests of other classes. Some classes of requests move through the network with a closed topology, i.e., external arrivals and departures are not allowed, other classes of requests move according to an open topology. The tagged value `PAserviceTime` specifies the average amount of time that requests spend at a service center. Mixed QN are generated from Use Case diagrams with multiple actors, and where each actor represents one user class, depending on its stereotype.

3.3 The transformation algorithm

The proposed algorithm, named Algorithm UML-QNE, translates an annotated UML specification into the QN

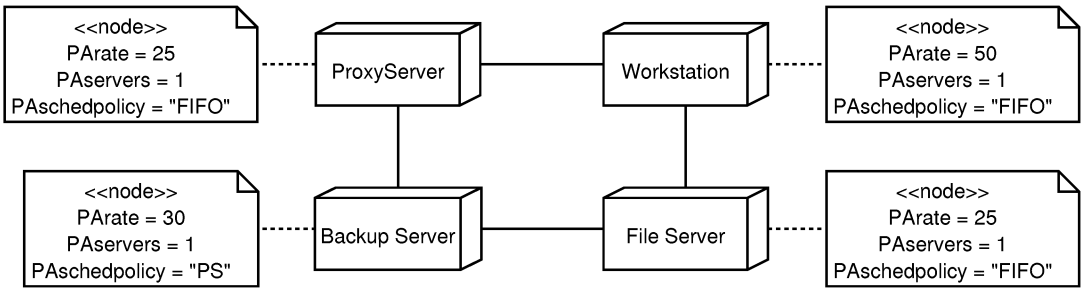


Figure 3: Example of annotated Deployment diagram

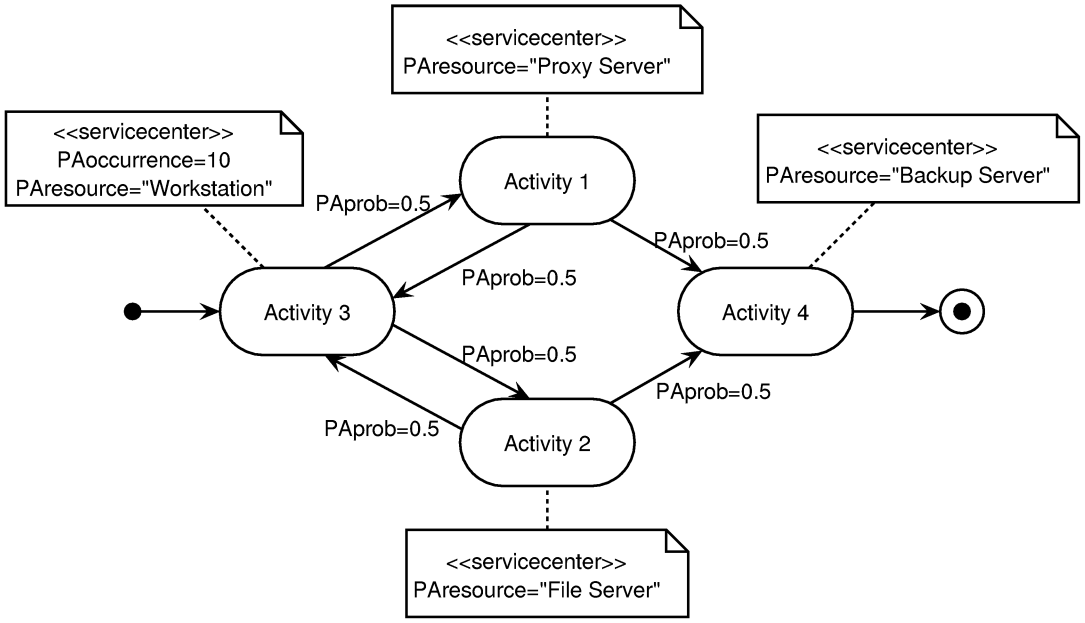


Figure 4: Example of Annotated Activity Diagram

model. We use the model notation shown in Table 1. In the algorithm we denote with *TagName*(X) the value of tag *TagName* associated to UML element X.

N	Number of nodes in the UML Deployment diagram
S_i	i-th service center
μ_i	Service rate of service center S_i
λ_i^C	Arrival rate of class C customers at service center S_i
NS_i	Number of servers in service center S_i
$\mathbf{P}^C$	$N \times N$ routing matrix for class C customers

Table 1: Notation used in Algorithm 1

The algorithm is illustrated in Fig. 1 and works according to the following steps:

1. For each Deployment diagram node we define a corresponding service center; if N is the number of nodes in the Deployment Diagrams, we define N service centers $S_1, S_2, \ldots S_N$.

2. For each Actor in Use Case diagrams we define a class of customers in the QN.

3. For each Activity diagram associated to an Actor we define the routing matrix for the current class of customers according to the predecessor-successor relationship of Action states. Routing probabilities are obtained from the `PAprob` tag associated to UML transitions.

The computational complexity of Algorithm 1 is $O(N + T + A)$, where N is the number of nodes in the Deployment Diagrams, T and A are the number of transitions and action states in all the Activity diagrams, respectively.

4 IMPLEMENTATION

A prototype Java tool called UML Queuing Network Evaluator has been implemented in order to demonstrate the approach described in the previous section. The tool parses an XMI representation of an annotated UML model to derive the QN model. The performance model is then solved using the MVA algorithm (Reiser and Lavenberg, 1980).

The internal structure of UML-QNE is shown in Fig. 6. UML-QNE builds an internal representation of the whole performance model. The QN model is made of resources and users (requests). Each user is associated to a `Topology` object, representing the routing matrix which applies to that user class. There are two different kind of users, which are `OpenUser` and `ClosedUser`; they describe open and

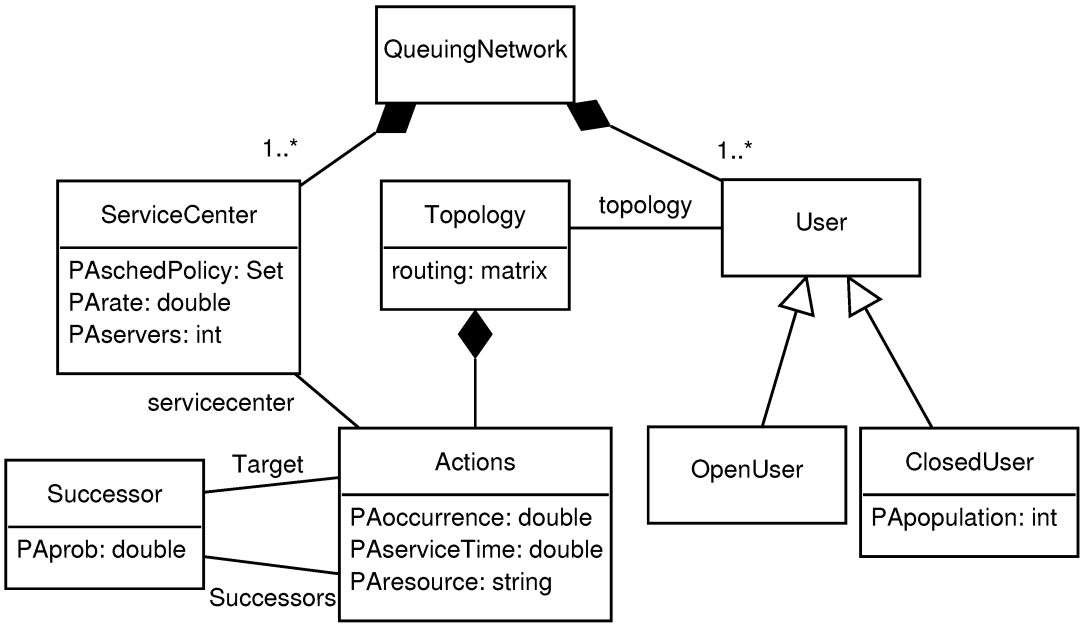


Figure 6: Class diagram of the UML-QNE Java tool

Algorithm 1 QN Model Generation

for all Deployment diagram node R_i, $i = 1 \ldots N$ **do**
  $S_i :=$ New Service Center
  $\mu_i := 1/PAserviceTime(R_i)$
  $NS_i := PAservers(R_i)$
end for
Let $C \leftarrow 0$
for all Actor A **do**
  Initialize routing matrix $\mathbf{P}^C$ for class C to zero
  $AD :=$ Activity diagram associated to A
  for all Transition t from action state a_i to a_j of Activity diagram AD **do**
    $R_k := PAhost(a_i)$
    $R_l := PAhost(a_j)$
    $P^C_{k,l} := PAprob(t)$
  end for
  if A is a ClosedUser **then**
    Set class C as a closed chain with $PApopulation(A)$ requests
  else
    Set class C as an open chain
    for all Action state a of Activity diagram AD **do**
      if $PAoccurrence(a)$ is defined **then**
        $R_i := PAhost(a)$
        $\lambda^C_i := PAoccurrence(a)$
      end if
    end for
  end if
  Let $C \leftarrow C + 1$ {New Customer Class}
end for

closed user classes, respectively. Each user can move through the service centers according with its routing matrix, which is described by an Activity diagram. A `Topology` object is made of a number of `Action` objects. Each `Action` is associated to a `ServiceCenter` node, representing the fact that the given action requests service to its associated service center. Actions are linked in a predecessor-successor relationship, each transition having an associated probability.

UML-QNE currently understands the UML models produced by the ArgoUML CASE tool (ArgoUML). ArgoUML model encoding is based on XMI (Object Management Group (OMG), 2002b), which is an XML-based notation for UML model representation. UML-QNE parses the XMI file and builds an internal representation of the UML model. This internal representation, whose structure is illustrated in Fig. 6, is then used to build the QN model, which is finally solved by using the MVA algorithm. Performance results include, for each service center S_i, the mean number of customers (mean queue length N_i), the utilization U_i, throughput X_i, and the mean waiting time R_i. Given that each service center in the QN model represents one resource in the UML model, performance results can be directly interpreted at the software architectural level.

5 AN EXAMPLE

The QN model of Fig. 7 is an example of a network derived from the Deployment and Activity diagrams of Fig. 3 and 4, respectively, assuming that the Activity diagram is associated to an actor stereotyped as $\ll$OpenUser$\gg$.

In this example, there is only one job class, which corresponds to the only actor in the UML model. The QN is made of four service centers $S_1, \ldots S_4$ which correspond to the four resources represented in the Deployment diagram of Fig. 3. S_1 corresponds to the "Proxy Server" node, S_2 corresponds to the "File Server" node, S_3 corresponds to the "Workstation" node and S_4 corresponds to the "Backup Server" node. The topology of the QN is derived from the Activity diagram of Fig. 4: each UML transition from activity i to activity j is mapped into an edge from server S_i to

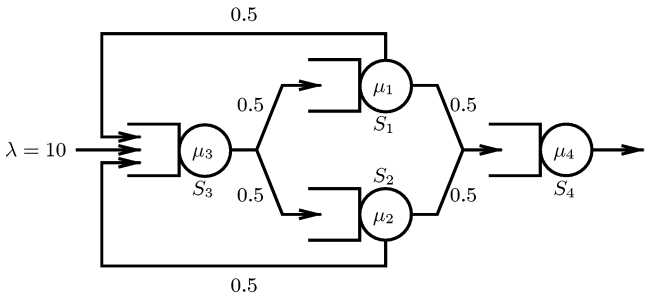


Figure 7: Open QN. Service rates are $\mu_1 = \mu_2 = 25$, $\mu_3 = 50$, $\mu_4 = 30$

server S_j. The service rates $\mu_1, \ldots \mu_4$ are derived from the `PArate` tags of the Deployment diagram.

The analysis of the product-form QN of Fig. 7 provides a set of average performance indices, that include mean number of requests, component utilization and throughput and average response time. Table 2 shows some numerical results for the specified set of parameters value, for each system component. The performance analysis can provide indication for possible system modification and further software performance evaluation can be iterated by choosing a different set of parameters value, to be inserted in the UML annotation.

Resource	N_i	U_i	X_i	R_i
Workstation	0.67	0.4	20.0	0.03
File server	0.67	0.4	10.0	0.069
Proxy Server	0.67	0.4	10.0	0.069
Backup Server	0.6	0.33	10.0	0.05

Table 2: Performance Results for the QN of Fig. 7: Mean number of Customers (N_i), Utilization (U_i), Throughput (X_i) and Mean Response Time (R_i)

6 CONCLUSIONS

In this paper we proposed an approach for performance modeling of UML software architectures. The approach considers annotations based on the UML Performance Profile, and derives a performance model based on Queuing Network. Performance evaluation of the QN by efficient algorithms provides a set of steady-state performance indices that characterize the software system behavior. Such results are immediately reported back in the UML software specification model as tagged values in the diagrams. The approach has been illustrated by implementing a prototype Java tool which automates the performance model derivation and solution phases.

Future works include the integration of UML-QNE in a more general framework for quantitative analysis of Software Architectures based on different techniques such as including simulation (Marzolla, 2004).

ACKNOWLEDGMENTS This work has been partially supported by MIUR research project FIRB "Performance Evaluation of Complex Systems: Techniques, Methodologies and Tools".

REFERENCES

ArgoUML. ArgoUML – Object-oriented design tool with cognitive support. http://www.argouml.org/.

L. B. Arief and N. A. Speirs. A UML tool for an automatic generation of simulation programs. In Proceedings of WOSP 2000 , pages 71–76.

S. Balsamo, A. D. Marco, P. Inverardi, and M. Marzolla. Experimenting different software architectures performance techniques: A case study. In Proceedings of WOSP 2004 , pages 115–119.

S. Balsamo, A. D. Marco, P. Inverardi, and M. Simeoni. Model-based performance prediction in software development: a survey. *IEEE Transactions on Software Engineering*, 30(5), May 2004b.

S. Bernardi, S. Donatelli, and J. Merseguer. From UML sequence diagrams and statecharts to analysable Petri net models. In Proceedings of WOSP 2002 .

V. Cortellessa and R. Mirandola. PRIMA–UML: a performance validation incremental methodology on early UML diagrams. In Proceedings of WOSP 2002 .

M. De Miguel, T. Lambolais, M. Hannouz, S. Betgé-Brezetz, and S. Piekarec. UML extensions for the specifications and evaluation of latency constraints in architectural models. In Proceedings of WOSP 2000 , pages 83–88.

H. Gomaa and D. A. Menascé. Performance engineering of component-based distributed software systems. In *Performance Engineering – State of the Art and Current Trends*, volume 2047 of *Lecture Notes in Computer Science*, pages 40–55. Springer, 2001.

V. Grassi and R. Mirandola. Towards automatic compositional performance analysis of component-based systems. In Proceedings of WOSP 2004 , pages 59–63.

G. Gu and D. C. Petriu. XSLT transformation from UML models to LQN performance models. In Proceedings of WOSP 2002 .

J. Hillston and Y. Wang. Performance evaluation of UML models via automatically generated simulation models. In S. A. Jarvis, editor, *UK Performance Engineering Workshop*, pages 64–78, July 2003.

P. Kähkipuro. UML-based performance modeling framework for component-based distributed systems. In R. R. Dumke, C. Rautenstrauch, A. Schmietendorf, and A. Scholz, editors, *Performance Engineering*, volume 2047 of *Lecture Notes in Computer Science*. Springer, 2001. ISBN 3-540-42145-9.

P. J. B. King and R. J. Pooley. Derivation of Petri net performance models from UML specifications of communications software. In B. R. Haverkort, H. C. Bohnenkamp,

and C. U. Smith, editors, *Computer Performance Evaluation / TOOLS*, volume 1786 of *Lecture Notes in Computer Science*, pages 262–276. Springer, 2000. ISBN 3-540-67260-5.

L. Kleinrock. *Queueing Systems*, volume 1. J. Wiley, 1975.

S. S. Lavenberg. *Computer Performance Modeling Handbook*. Academic Press, New York, USA, 1983.

C. Lindemann, A. Thümmler, A. Klemm, M. Lohmann, and O. P. Waldhorst. Performance analysis of time-enhanced UML diagrams based on stochastic processes. In Proceedings of WOSP 2002 .

M. Marzolla. *Simulation-Based Performance Modeling of UML Software Architectures*. PhD Thesis TD-2004-1, Dipartimento di Informatica, Università Ca' Foscari di Venezia, Italy, Feb. 2004.

Object Management Group (OMG). Unified modeling language (UML), version 1.4, Sept. 2001.

Object Management Group (OMG). UML profile for schedulability, performance and time specification. Final Adopted Specification ptc/02-03-02, OMG, Mar. 2002a.

Object Management Group (OMG). XML Metadata Interchange (XMI) specification, version 1.2, Jan. 2002b.

D. C. Petriu and H. Shen. Applying the UML performance profile: graph grammar-based derivation of LQN models from UML specifications. In T. Field, P. G. Harrison, J. Bradley, and U. Harder, editors, *Computer Performance Evaluation / TOOLS*, volume 2324 of *Lecture Notes in Computer Science*. Springer, 2002. ISBN 3-540-43539-5.

R. J. Pooley and P. J. B. King. The Unified Modeling Language and performance engineering. In *IEE Proceedings – Software*, volume 146, pages 2–10, February 1999.

Proceedings of WOSP 2000. *Proc. of the Second International Workshop on Software and Performance (WOSP 2000)*, Ottawa, Canada, Sept. 2000. ACM Press.

Proceedings of WOSP 2002. *Proc. of the Third International Workshop on Software and Performance (WOSP 2002)*, Rome, Italy, July 24–26 2002. ACM Press.

Proceedings of WOSP 2004. *Proc. of the Fourth International Workshop on Software and Performance (WOSP 2004*, Redwood Shores, California, Jan. 14–16 2004. ACM Press.

M. Reiser and S. S. Lavenberg. Mean value analysis of closed multichain queueing networks. *Journal of the ACM*, 27: 313–322, 1980.

C. U. Smith. *Performance Engineering of Software Systems*. Addison-Wesley, 1990.

C. U. Smith and L. Williams. *Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software*. Addison-Wesley, 2002.

J. Xu, C. M. Woodside, and D. C. Petriu. Performance analysis of a software design using the UML profile for schedulability, performance and time. In P. Kemper and W. Sanders, editors, *Proc. of 13th International Conference on Modelling Techniques and Tools for Computer Performace Evaluation - Performance TOOLS'2003*, volume 2794 of *Lecture Notes in Computer Science*, pages 291–307. Springer, Sept. 2003.

PERFORMANCE EVALUATION OF OCDR MECHANISM USING NON-MARKOVIAN SPN

Mohamed Escheikh[*], Jouda Maatoug[*] and Kamel Barkaoui[**]
[*] Lab. SysCom ENIT, Tunis
[**]CEDRIC-CNAM, Paris
Email: barkaoui@cnam.fr

KEYWORDS
Non-Markovian stochastic Petri nets, ATM connection management, Supplementary variable method.

ABSTRACT

In this paper we present a performance study of a connection management mechanism in ATM network namely OCDR (On-demand Connection with Delayed Release).
This study is based on a Stochastic Petri net model with general firing time distributions. The performance analysis is obtained by the method of supplementary variables implemented in SPNica tool which is based on Mathematica .

1. INTRODUCTION

The use of stochastic Petri nets (SPNs) is popular for performance evaluation of distributed communications systems. However, due to complexity of numerical state-space-based analysis methods, SPNs are often restricted to Markovian hypothesis (exponential transitions). In order to increase the modeling power, especially in telecommunication field, the consideration of non memoryless distributions seems to be ineluctable (German 2000a) (Heindl and German 2001). Recently, significant research efforts have been devoted to develop analysis techniques for SPN with general firing time transitions (general transitions) (German 2000a), (Haverkort 1995). The analysis approach used in this paper relies on the of method of supplementary variables (Cox 1955) (German and Lindemann 1994) which allows system dynamics description by differential equations and is implemented in SPNica analysis tool (German 2000b).
The main restriction of this approach is that activities (transitions) with non-exponentially distribution duration are mutually exclusive, i.e., in each marking at most one general transition may be enabled. This leads to reasonable costs algorithms to generate vector states.
The performance study presented in this paper concerns the optimization of the bandwidth utilization for a given traffic source distribution in ATM networks under the OCDR mechanism.
ATM networks give guaranteed QoS through resources reservations, call admission control, etc. Asynchronous transfer mode is a connection oriented network using packet switching. A connection is established between source and destination before packet transmission. Connection setup accumulates a certain delay. We distinguish two main types of connection management in ATM: connection per packet (CpP) and permanent connection (PC). The CpP is considered as a connectionless service whereas PC is a connection oriented one.
Given the inefficiency and the low utilization ratio of CpP and PC for variable and sporadic traffic, the OCDR mechanism is suggested in order to improve throughput performance.

The connection is maintained after the packet transmission for a certain delay called holding time (HT). The connection is released if no new packet arrives during the holding time.
This paper is organized as follows. The supplementary variable is briefly described in section 2. The OCDR Non-Markovian SPN model is detailed in section 3. Section 4 is devoted to performance analysis of OCDR mechanism, before concluding the paper in section 5.

2. NON-MARKOVIAN SPNs

Large application classes in communication systems have been analyzed using stochastic Petri net under Markovian hypothesis. In Markovian SPNs all timed transitions have an exponentially distributed firing time or a geometrically distributed firing time. This leads to a Markov chain as an underlying stochastic process. Non-Markovian SPNs includes any SPN which is not Markovian. In DSPNs (Ajmone Marsan and Chiola 86) the timed transition are either exponential or deterministic and the deterministic transitions are mutually exclusive and have preemption policy prd (preemption repeat different). In MRSPNs (Markov Regenarative SPN) (Choi et al. 1994) (Ciardo et al. 1994), the stochastic process is Markov regenerative and general transitions (including the deterministic ones as a special case) with prd-policy are allowed. We focus in this paper on supplementary variables method for the stationary analysis of Non-Markovian SPNs.

2.1 Supplementary Variables Method

The method of supplementary variables (German and Lindemann 1994) is one of three main analysis techniques for Non-Markovian SPNs with phase method and embedding method. It deals with Non-Markovian SPNs dynamics where general transitions are mutually exclusive.

The principle of this method consists in adding continuous variable to the discrete state description representing the status of the timed activities. This discrete and continuous state process represents all relevant information about the history and is therefore a Markov process. The Supplementary variable method is implemented in the prototype tool SPNica (German 200b) used in this study to quantify the influence of main parameters (in particular the holding time parameter) on the performance of OCDR mechanism.

3. OCDR MODEL

The OCDR mechanism has been analyzed in (Haverkort 1995) by infinite stochastic Petri nets (iSPNs) using matrix-geometric solution under an On/Off source trafic. A continuous time Markov chain (CTMC) is also given. iSPN circumvent large state space problem with one unbounded place and all transitions considered as exponential. In (German 2000) the same model is extended to consider mutually exclusive general transitions.

3.1 Model Description

The model given in Figure 1 is based on (German and Telek 1999). In order to remove conflict between the transitions start and release the former is given a priority equal to one. For more details see (German 2000a.).

4. PERFORMANCE EVALUATION OF OCDR MECHANISM

We present in this paragraph the input and output parameters of the OCDR model described above.

The OCDR model input parameters are:

- Firing rate of *exponential* transition arrival: $\lambda = 100$ pkts/s
- Firing time distribution of transition service: *uniform* from a = 0.00008 to b=0.0016

- Firing time distribution of transition connect: *deterministic* time τ^c=0.01s
- Firing time distribution of transition release **:** *deterministic* time τ^r= 0.02s
- Buffer capacity K: variable

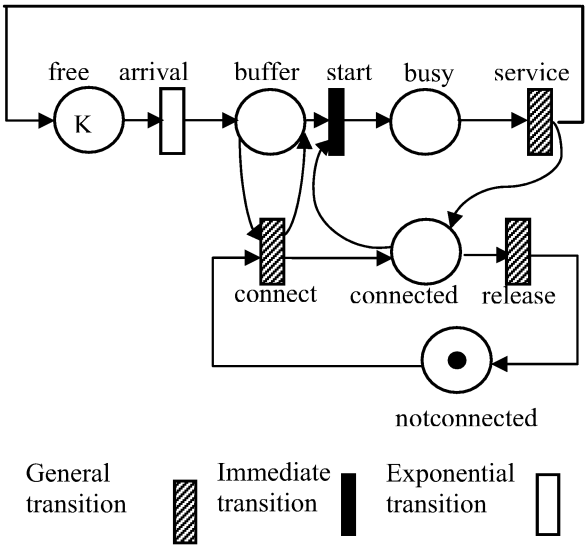


Figure1: Non Markovian SPN OCDR model

The OCDR model output parameters are evaluated through the following measures:

- N = E [#buffer + #busy]: the mean number of packets in the system.
- S = E [# service]: the system throughput.
- W = N/S: the mean waiting time according to little law (in seconds).
- BWused = $E[1_{\{\#busy=1\}}]$: the fraction of actually used bandwidth.
- BWreq = $E[1_{\{\#notconnected=0\}}]$: the fraction of requested bandwidth.
- U = BWused/ BWreq: the utilization of requested bandwidth.
- Sc=Mean rate of established connections

N, BWused, and BWreq are expected rate rewards and S is the expected impulse reward.

Mean Waiting Time versus Mean Traffic Arrival Rate λ

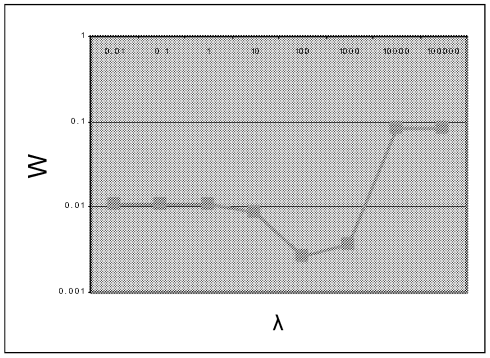


Figure 2: the mean waiting time versus λ

Interpretation

Phase 1: in this phase the waiting time in the system can be practically resumed to the connection establishment time.

Phase 2: W decreases significantly. In fact packets arrive faster (λ increase), they are transmitted whereas connection is established (during the holding time).

Phase 3: W increases to a threshold corresponding to saturation, the augmentation of the traffic rate λ does not affect waiting time in the system given a finite buffer capacity.

Utilization of Requested Bandwidth versus Time Release

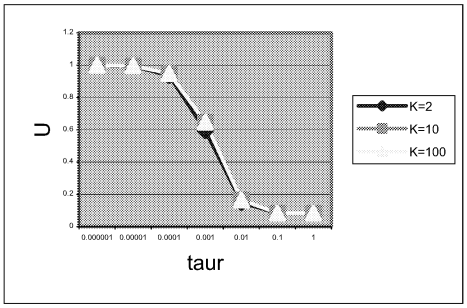


Figure 3: the utilization of requested bandwidth as a function of the time release

Interpretation: The utilization of requested bandwidth as a function of the holding time is given in figure 3. As the holding time (HT) increases, U increases. In fact, the HT corresponds to a time during which the connection is maintained without being used. HT increases, connection is maintained without being used and used bandwidth decreases.

Bandwidth used versus Time Release

In figure4 we depict the bandwidth used as a function of the time release for a given λ.

Interpretation:
For K=10 and K=100 the bandwidth used is practically constant versus the time release. However for K=2 the bandwidth used increases until reaching the same value for K=10 and K=100.

This can be interpreted as follows: increasing the holding time (HT) has the same effect on the bandwidth used as increasing buffer capacity.

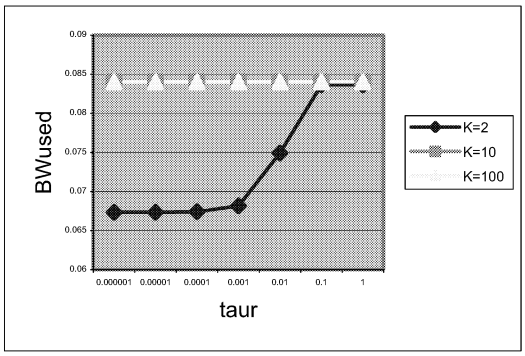


Figure 4: the bandwidth used as a function of the time release

Bandwidth Requested versus Time Release

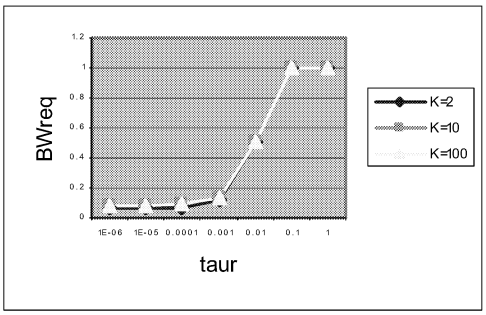


Figure 5: the bandwidth requested as a function of the time release

Interpretation:
The bandwidth requested increases as the time release increases until a saturation value (Figure5). Each established connection will be maintained more time at least equal to the holding time. Note that increase of BWreq versus taur is more significant for a given interval.

Mean Service Rate versus Time Release

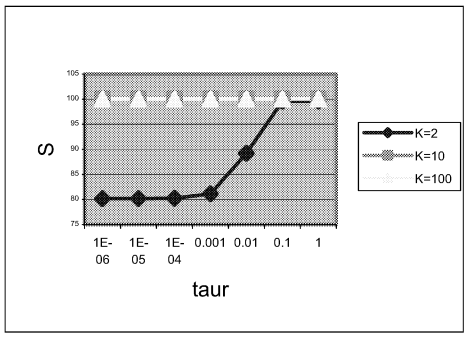


Figure 6: Throughput S versus the time release

Interpretation:
The figure 6 shows the evolution of the throughput S versus the time release for different buffer capacities K. For K=10 and K=100 the throughput is practically constant versus the time release. However for K=2 the bandwidth used increases until reaching the same value for K=10 and K=100. Increasing the holding time has the same effect on the throughput as increasing buffer capacity.

Mean Number of Packets in the System versus Time Release

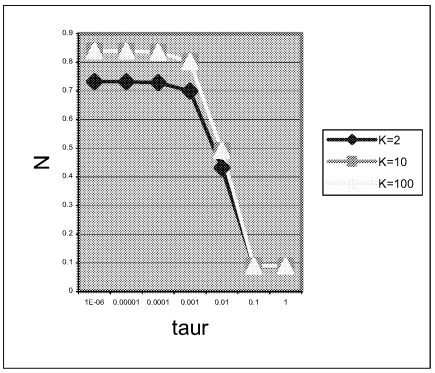


Figure 7: the mean number of packets in the system versus the time release

Interpretation: The mean number of packets in the system (waiting for service or being effectively served) decreases when the time release increase (figure 7).

Mean Connection Established Rate versus Time Release

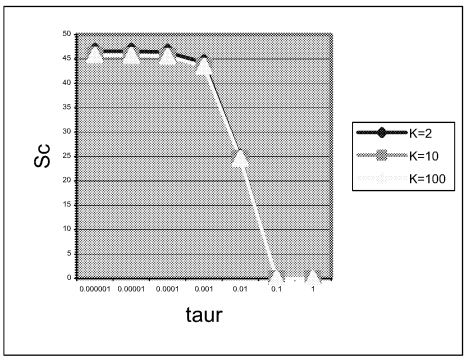


Figure 8: Mean connection established rate versus the time release

Interpretation: Mean connection established rate decreases versus time release until zero for taur=0.05 for different value of K. Figure 8 depict three phases. The first and the third ones correspond to an Sc constant whereas the second phase includes a significant impact of the time release on Sc.

5. CONCLUSION

A performance study of the OCDR mechanism based on SPN model with general firing time distributions has been presented. The analytical results were performed with SPNica tool where the method of supplementary variables is employed. This study allow to quantify the impact of the variation of holding time (time release) and suggests that OCDR is particularly efficient for a given holding time interval. Out of this interval OCDR behaves either like connection per packet (CPp) or permanent connection (PC) for a given traffic distribution.

REFERENCES

Ajmone Marsan M; and G. Chiola. 1986. "On Petri nets with deterministic and exponentially distributed firing times" *Lecture notes in Compuetr Science, vol. 266, Springer, Berlin, pp. 132-145.*

Ciardo, G.; R. German; and C. Lindemann. 1994. "A characterization of the stochastic process underlying a stochastic Petri net" *IEEE Trans. Software Eng*. 20, pp 506-515.

Choi, H; V.G. Kulkarni; and K.S. Trivedi. 1995. "Markov regenerative stochastic Petri nets" *Performance Evaluation 24 pp 47-68.*

Cox, D.R. 1955. " The analysis of non-Markov stochastic process by the inclusion of supplementary variables" *Proc. Camb. Phil. Soc. 51, 433-441.*

German R; and C. Lindemann 1994. "Analysis of stochastic Petri nets by the method of supplementary variables" Performance Evaluation 20, pp 317-335.. 1999. "Formal relation of Markov renewal theory

German R; and M. Telek. 1999. "Formal relation of Markov renewal theory and supplementary variables in analysis of stochastic Petri nets". *In Proc. 8th int. Workshop on Petri Nets and Performance Models*, pages 64-73. Zaragoza , Spain.

German, R. 2000a. "Performance analysis of communication systems: modelling with non-Markovian stochastic Petri nets". *1st ed. John Wiley and Sons*,.

German, R. 2000b. "Markov regenerative stochastic Petri net with general execution policies: supplementary variable analysis and a prototype tool" *Performance Evaluation, 39, pp 165-188.*

Haverkort B.R 1995."Matrix-geometric solution of infinite stochastic Petri nets". *In Proc.IEEE Int. Performance and Dependability Symp.* pp 72-81.

Heindl, A; and R German,. 2001. "Performance modeling of IEEE 802.11 wireless LANs with stochastic Petri nets". *Performance Evaluation*. 44 pp 139-164.

MODELING DISTRIBUTED SUPERVISORY SYSTEMS USING A MODULAR MULTI-FORMALISM METHODOLOGY

Mauro Iacono [**] **Stefano Marrone** [***]
Nicola Mazzocca [****] **Francesco Moscato** [†]

[*] *Seconda Universitá di Napoli, Aversa, Italy*
[**] *mauro.iacono@unina2.it*
[***] *stefano.marrone@unina2.it*
[****] *nicola.mazzocca@unina2.it*
[†] *francesco.moscato@unina2.it*

KEYWORDS Multi-Formalism Modeling, Supervisory System, Real-Time Requirements, Colored Petri Nets, Stochastic Well Formed Petri Nets, Queuing Networks

ABSTRACT

Requirements of modern supervisory systems well match with the flexibility offered by off-the-shelf computing devices, and cost reasons justify the use of these devices in such systems. In order to obtain effective implementations, system tasks can be distributed over several computing nodes, to separate system functions and better cope with complexity or to adapt the architecture to the plant distribution. Consequently, distributed heterogeneous systems are now a more and more adopted solution in this field in order to support these requirements.
Resulting architectures are versatile and multi-layered but need enhanced modeling techniques to cope with their complexity. In this paper a multi-formalism approach is presented, based on Stochastic Well-formed Petri Nets and queuing networks, suitable for performance design and evaluation of heterogeneous real-time supervisory systems. The approach is applied to a real case study.

INTRODUCTION

Heterogeneity in complex Distributed Supervisory and Control Systems is a key issue in design and performance assessment and evaluation. A multi-formalism approach to model system behavior allows different components to be described by using the most suitable modeling formalism since no single one is adequate to characterize all system parts and features. Moreover, a modular approach helps to cope with complexity of the modeling and evaluation process using a divide-et-impera strategy. Petri Nets (PN) and Queuing Networks (QN) are widely used to estimate both hardware and software performances of complex systems (Marie and Marie, 1993) and (Marsan *et al.*, 1986). PN are used for qualitative (and in some evolutions also quantitative) analysis whereas QN are employed for performance evaluation (quantitative analysis). Only few efforts have been undertaken in the past to combine description features of QN and PN ((Lin and Fu, 1999), (Bause, 1993), (Balbo *et al.*, 1988)). In particular, in ((Lin and Fu, 1999), (Bause, 1993)) the original Generalized Stochastic Petri Nets (GSPN) formalism was extended by "hiding" queues in special places. This approach allows an analysis of the model of the whole system by using a single formalism in spite of an increase of complexity in the model. In a previous work (Moscato *et al.*, 2003), we proposed a methodology based on Multi-Formalism approach, showing its effectiveness on a robotized cell case study. Although the results were very co-

herent with experimental data, the modeling formalisms we used did not allow anymore to capture all the system characteristics, such as priorities both in network traffic and in operating system schedulers. In this work we improve our existing multi-formalism methodology to include also the possibility of modeling priorities in both network traffic and task scheduling. This is supported by the adoption of Stochastic Well-formed Petri Nets (SWN) and Queuing Networks (QN) with priority as modeling languages in order to capture node diversity, task priority, operating system scheduling and network behavior in a heterogeneous, distributed supervisory system. We show the effectiveness of our improved modeling methodology by applying it in the design process of the new supervisory system for the robotized cell. Model results are validated by a comparison with real system measures to show both correctness and preciseness.

CASE STUDY

In modern automated systems it is possible to individualize different stacked control layers each one in charge of specific activities and responsibilities. According to (Miyagi *et al.*, 2001) a *Supervisory System* is defined inside a hierarchical structure as shown in 1. On the bottom of this

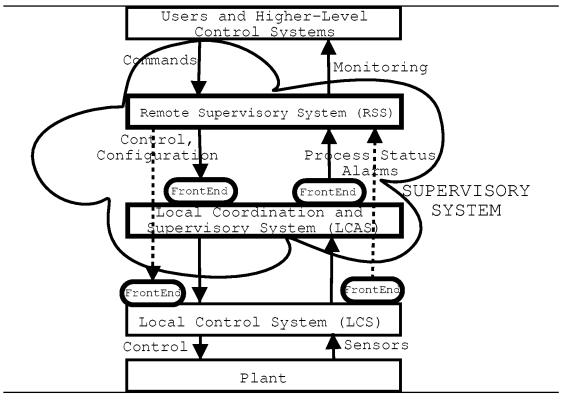


Fig. 1. System Architecture

stack is the plant layer; it is composed of all the non-computing devices like sensors and engines. On the top of this layer there is the Local Control System (LCS) level that is composed of local analog and digital controllers that operate directly on the plant. Distribution of the plant and of the local control, due to infrastructural needs, safety reasons or for performances, makes necessary the presence of an higher layer which coordinates local controllers in order to best perform the control tasks; this layer is named Local Coordination And Supervision (LCAS). On the top of this level a (Remote) Supervisory System (RSS) is necessary for safety and monitoring purpose. This control is usually performed by general purpose PC, connected to LCAS through a computer network or dedicated devices which communicate directly to LCS layer controller.

The automated system studied in this paper is a robotized cell (see Figure 2) composed of two

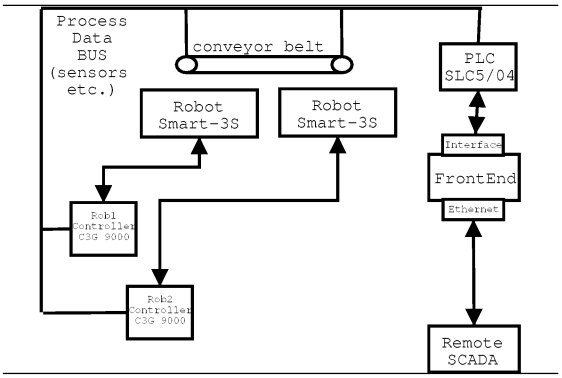


Fig. 2. SCADA Architecture

robots with related control units, a conveyor belt and some sensors, controlled by an Allen-Bradley PLC to perform a palletization process. The PLC serves a SCADA system composed of two computing nodes (commonly available, off-the-shelf PCs): the first one, namely FrontEnd, is directly interfaced to the PLC and implements the local supervision; the second, namely SCADA, is used by the human supervisor through a network in order to manage the cell from his office. The two nodes exchange control, status and configuration data. Supervisory application tasks run on the nodes to perform the system mission. Tasks are scheduled with priority and use the network (a TCP/IP link) with the same priorities, induced by tasks (the link is dedicated). The goals of performed analysis are to locate system bottlenecks and to provide parameters to be used to tune the system in order to fulfill real-time requirements. Further details are given in the following and in (Moscato *et al.*, 2003).

MOTIVATION AND BACKGROUND

The main problem in design of distributed remote control and supervisory systems is the relevance of time in communication and computations. The presence of alarms or critical tasks introduces real-time constraints and dependability (and correctness) requirements. We are interested here in capturing system performances in order to help the designer in both early design and validation phases giving information about real time requirements verification and overall system loads and bottlenecks. In our previous work (Moscato *et al.*, 2003), we proposed a methodology aimed to model the computing nodes in a distributed su-

pervisory system, their load in terms of operating system and application tasks and communications between nodes in presence of concurrency and parallelism among tasks and shared resources.
The main contribution of this paper is the improvement of the methodology explained in (Moscato *et al.*, 2003) to allow to model priorities classes of tasks and network communications. The formalisms by which the system is modeled in the proposed methodology are Stochastic Well-formed Nets (SWNs) and Queuing Networks (QNs) with priority. The first ones are derived form a particular class of high level Petri nets, the Well-formed Nets (WN), adding the stochastic distribution on the transitions firing rates. On the other side, WNs represent an improvement of CPNs in order to enhance the complexity of the analysis. In fact they are substantially identical in their expressive power; the main difference besides on the syntactic definition which gives the WNs the ability to be analyzed according to the symbolic marking rule. This leads to the construction of the Symbolic Reachability Graph, a more efficient way to represent the state space of the model. QNs are a graph-based formalism, too; through the use of two different kind of nodes (servers and queues), they are used to describe the flow of the information processed in a computing system in order to obtain stochastic description of system performances. The addition of priority classes allows the formalism to be a more flexible modeling tool and to cope with the issues of most real world systems, such as OS schedulers. A formal and complete discussion on the formalisms can be found in (Chiola *et al.*, 1993)

MODELING APPROACH

In this paper we stress the importance of priority level design in both nodes and network, with consequent impact on the modeling process. The individualization of priority classes is performed with reference to tasks, while the network traffic priority is derived from priorities of the tasks that generate it. Task priorities are defined by examining system specifications.
The methodology we propose is based on the separation of the system to be modeled in two parts (submodels),each one of those is modeled with a proper formal technique: Supervisory Applications and System Infrastructure.
The supervisory application tasks are in charge of pursuing individual services in the systems and rely on the node infrastructure to perform their actions: they are not aware of the rest of the load of the system and consequently influence each other in competing for resources.
The overall behavior of these tasks (in each separated priority level) can be represented by a set of processes which in turn compute, compete for resources and wait for conditions before continuing.
Infrastructure tasks on the other side are coordinated and allocated all together by the operating system, which also manages the application tasks by applying the chosen scheduling algorithm. The overall behavior of these task can be viewed as a group of customers which access a queue when requesting CPU or network.

The modeling process is organized in steps:

(1) Individualization of application tasks;
(2) individualization of priority classes for application tasks;
(3) task mapping on nodes.
(4) Individualization of OS tasks and communication tasks;
(5) individualization of priority classes for OS tasks.
(6) Measurement/characterization of service time for isolated application tasks for each priority level;
(7) measurement/characterization of response times for OS tasks for each priority level, accounting for scheduling policy and for communications.

Step sets 1-3 and 4-5 constitute the inspection phase of the methodology. They can be pursued in parallel and are propaedeutic for models themselves. The first group is related to the specific supervisory application, the second to the infrastructure of the system. Steps 6-7 constitute the model synthesis phase of the methodology, and are pursued in sequence by using SWN and QN respectively. While steps 1-5 consist in a qualitative analysis of system requirements and specifications, steps 6 and 7 are the means by which specifications can be turned in quantitative evaluation of the model.

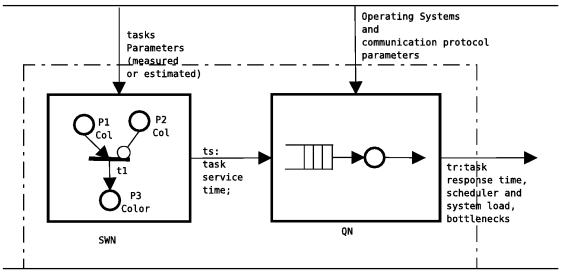


Fig. 3. SWN and QN submodels interactions

To better cope with both qualitative and quantitative analysis of each part of the system and also to allow submodels reuse a modular multi-formalism approach is used to model the whole system. The methodology is shown in Figure 3. Both for SCADA and Front-End subsystems, an

SWN model is used to model tasks behaviors on each node, while a QN model is used to take in account of operating system and network behaviors. Some measures retrieved from some stub implementations of system parts are used to tune the SWN and QN models. For further informations refer to (Moscato *et al.*, 2003)
Although SWN could be used also to describe queuing effects, we used both QN and SWN formalism for several reasons.

First of all QN formalism is a *native* formalism to describe queuing effects; in addition SWNs do not provide native mechanisms to establish the average load of a service. Finally the SWN suffers of state space explosion problems (Murata, 1989) so it could be a good choice to reduce the complexity of these models by using a more appropriate formalism to model some aspect of the system.
SWN and QN submodels interacts by exchanging parameters. SWN submodels, which model behavior of control and supervisory tasks and interface drivers accounting for concurrency among tasks, real-time constraints and priorities effects, are used to evaluate execution times of each task (called *tasks service times* in Figure 3 because of their use as service times in the QN model described later) as well as transmission request distributions of each task that require data transmission. Time needed to process computation in SWN submodels is obtained by using experimental results of test code run on the real target system.
QN submodels represents behavior of operating system and communication protocol, considering all tasks running concurrently or in parallel on each computing node (FrontEnds and Remote Supervisory Systems in Figure 1). Service time of scheduled tasks and their scheduling request distributions are provided by the tasks SWN submodel, as well as the queuing effects of transmission protocol such as TCP/IP. The QN submodel produces effective execution time of each task (*response time* in terms of QN) in the real load conditions of the system, as well as the characterization of system bottlenecks in task scheduling and network. Overall results can be used to tune the system in order to fulfill real-time constraints or to support the evaluation of the effects of variations in the number of supervisory tasks or in the frequency of transmission operations needed, for example, to read plant process variables from remote supervisors.

THE MODEL

The use of SWN greatly improves the modeling technique with respect to the Generalized Stochastic Petri Nets (GSPN) approach used in the methodology adopted in (Moscato *et al.*, 2003). The higher expressive power allows for inclusion of the new priority characterization in a compact way, and allows the application model to be efficiently represented with less nets. In fact nets exploit colors to represent priorities as well as different task behavior, thus giving an overall representation of the system application level.
Figure 4 shows the SWN model for the application

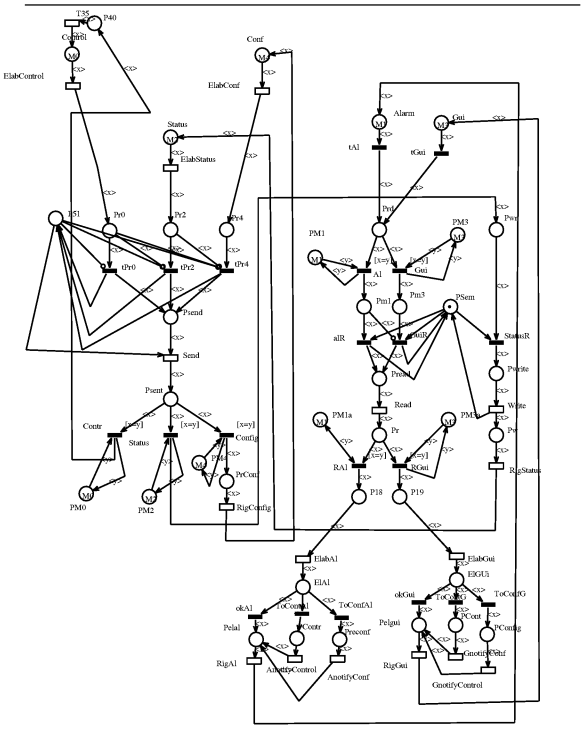


Fig. 4. SWN Model for the SCADA node

level of the SCADA node. The SWN model of the FrontEnd is omitted for brevity sake.

In the case study, 5 tasks are allocated on the node, each at a different priority level. This is modeled by using different tokens in *Control*, *Status*, *Conf*, *Alarm* and *GUI* places in Figure 4. Tasks are scheduled on a priority, non-preemptive round-robin scheme. The application is responsible of managing an overall number of 46 field variables (26 status, 20 alarms) at least once in a second. Control signals must be delivered with a maximum delay time of 0.3 seconds, while configuration signals have no real time specification. The Aim of the modeling process is to assure that these design constraints will be satisfied during the normal system activity.

The 5 tasks lead to 5 different priority levels:

(1) control signals management;

(2) alarm management;
(3) state management;
(4) Graphical User Interface (GUI);
(5) configuration signals management.

The higher is the number, the lower is the assigned priority. Priorities are modeled through the definition of a Color Class with five colors. In addition, priority dependent task behaviors are modeled by appropriate arc functions. Tasks access in read/write mode a shared memory to perform their actions. In the Figure 4 read operations start at the P_{rd} place while write ones start at the P_{wr} place. Non-preemption is guaranteed by the presence of a semaphore P_{Sem}, which manages read/write conflicts when accessing the shared memory. With reference to Figure 4, on the left in the upper half the tasks for sending control signal (see *Control* place), sending configuration (see *Conf* place) and obtaining status (see *Status* place) are modeled.
On the left in the lower part of the figure the message sending process for the three tasks is depicted. These tasks communicate with the Front End node.
On the right, the shared memory access protocol (implemented by a semaphore construct) is described. Read operations are performed by the alarm conditions evaluation task and GUI task, which in turn are modeled on the right. A certain percentage of these tasks generate warnings for the human supervisor about the need for configuration and control signaling actions (see *ANotifyControl* and *ANotifyConf*, *GNotifyControl* and *GNotifyConf* transitions). Priorities are managed transparently by using transition guards and arc conditions and one color for each priority level. The model can also manage more than one task for each priority level (all transitions but *Read*, which has infinite-server semantic because reads can be overlapped, have single-server semantic). The infrastructure of the SCADA node is modeled

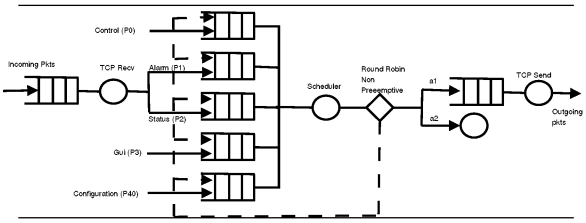


Fig. 5. QN Model for the SCADA node

by a priority QN (see Figure 5). The FrontEnd QN model is omitted for brevity sake.

The QN is composed by two sections: the first models the priority OS scheduler of the node, which is characterized by a non-preemptive round robin policy with priority; the second models the incoming and outgoing TCP/IP traffic management.

The first has 5 queues with different priorities and one server with 5 different service times (t^i_{serv}), obtained by solving the SCADA SWN model, and a quantum slice of 10 msecs. Serviced processes exit the queue, while the others are reenqueued, waiting for completion. In the following: - $t_{s_{(q,j)}}, j \in (0..4)$ is the service time of task slices belonging to a task with priority j that use a whole quantum slice;
- $t_{s_{(p,j)}}, j \in (0..4)$ are the service times of task slices belonging to a task with priority j that use less than a whole quantum slice;
- $p_{(q,j)}$ and $p_{(p,j)}$ are respectively the relative rates of tasks with service time $t_{s_{(q,j)}}$ e $t_{s_{(p,j)}}$;
- $t_{s_j} = t_{s_{(p,j)}} + n_j \cdot t_{s_{(q,j)}}$ where n_j is the number of full quantum slices needed by a task of priority j to be completed; - $\lambda_{(q,j)}$ and $\lambda_{(p,j)}$ are respectively the arrival rates of tasks with service time $t_{s_{(q,j)}}$ e $t_{s_{(p,j)}}$;
- $\rho_j = \sum_{k=0}^{j}(\lambda_{(q,k)} \cdot t_{s_{(q,k)}}) + (\lambda_{(p,k)} \cdot t_{s_{(p,k)}})$ is the system load generated by tasks with priority levels from 0 to j.

Arrivals distributions for the *Scheduler* service are experimentally determined by using stub implementation of the tasks requesting services on SCADA and FrontEnd. With these distributions as inputs of the scheduler service, it is possible to prove that the OS scheduler queue model, with the five priority queues shown in Figure 5, can be approximated by a single M/G/1 queue with:

$$t_s = \sum_{j=0}^{4}(t_{s_{(q,j)}} \cdot p_{(q,j)}) + (t_{s_{(p,j)}} \cdot p_{(p,j)}) \qquad (1)$$

$$\sigma_s^2 = \sum_{j=0}^{4}(t_{s_{(q,j)}} - t_s) \cdot p_{(q,j)} + (t_{s_{(p,j)}} - t_s) \cdot p_{(p,j)} \qquad (2)$$

$$C_s^2 = \frac{\sigma_s^2}{t_s^2} \qquad (3)$$

For each priority level, the sojourn time t_{r_i} can be evaluated as:

$$t_{r_i} = \frac{\sum_{j=0}^{4} \rho_j \frac{(t_{s_{(q,j)}} - t_s) \cdot p_{(q,j)} + (t_{s_{(p,j)}} - t_s) \cdot p_{(q,j)}}{2\lambda_j}}{(1-(\rho_0+\ldots+\rho_i))(1-(\rho_0+\ldots+\rho_{i-1}))} + t_{s_i}$$

The Service times of each task are evaluated by solving the SCADA SWN model of Figure 4. For Example, to evaluate the service time of the *Status* task, the inverse of the throughput of the transition *RigStatus* in Figure 4 was used and the service times arrival distribution was approximated as an exponential one with the value obtained from the SWN model used as the mean of arrival times distribution.

The prove is omitted for brevity sake. The rest of the model describes TCP/IP communications as two M/M/1 queues (*TCP Recv* and *TCP Send*). TCP Recv feeds the scheduler queue, while TCP

Send is fed by it. The absence of other traffic on the network guarantees that actual network traffic is generated according to tasks request, which priorities are inherited by packets.

EXPERIMENTAL RESULTS

The model of the case study obtained by applying the methodology has been used to simulate the system behavior. The model results were used to assure that, once the polling interval of each task was fixated, the real time specification described in section 5 were satisfied. Results can be compared with the values obtained with experiments performed on the real system. Results are in Table 1. Comparison shows that the model describes the system with great accuracy.

Table 1. Some Experimental Results

Parameter	Model	Value measured on real system	Variation
$t_{sControl}$	3.03	3.11	2.6%
$t_{sStatus}$	3.31	3.44	3.7%
$t_{sConfig}$	4.39	4.20	4.5%
$t_{rControl}$	14.85	15.34	3.2%
$t_{rStatus}$	16.23	17.04	4.7%
$t_{rConfig}$	18.37	17.48	5.1%

Results give a maximum variation from real system parameters of 5.1% for response times and 4.5% for service times, in both cases for the reconfiguration task. Such a behavior is due to the fact that configuration signal management is the lowest priority task, and consequently it is influenced more than tasks in the other priority levels by approximations in the QN model.

CONCLUSIONS AND FUTURE WORKS

In this paper we exploited the benefits of modularity to combine the advantages of Petri Nets and Queuing Networks. A multi-formalism approach allowed to model each logical part of the system with the most suited formalism, thus giving the modeler a way to better focus on solving the problem more than forcing a formalism to implement the solution. We showed that the evolution of the methodology towards priority representation is fully supported by the SWN formalism: moreover, the increased expressive power of such a conceptual tool also allows for more compact representations of the models. Color has been exploited in order to model different classes but also different tasks in the same class on the same net, thus allowing the whole case study to be represented by 2 nets, while in the non-colored case we needed 5 nets to model the problem, without priorities.
Future works will include the automation of modeling and solution process by means of the OsMoSys framework (Vittorini *et al.*, 2003).

REFERENCES

Balbo, G., S.C. Bruell and S. Ghanta (1988). Combining queueing networks and generalized stochastic petri nets for the solution of complex models of system behavior. *IEEE Transaction on Computers* **37**, 1251–1268.

Bause, F. (1993). Queuing petri nets - a formalism for the combined qualitative and quantitative analysis of systems. *Proc. of 5th International Workshop on Petri nets and Performance Models* pp. 14–23.

Chiola, G., C. Dutheillet, G. Franceschinis and S. Haddad (1993). Stochastic well-formed coloured nets for symmetric modelling applications. *IEEE Transactions on Computers* **42**(11), 1343–1360.

Lin, M.-H. and L.-C. Fu (1999). Modeling of priority queuing service in discrete event systems using hybrid petri nets. *Proc. of the IEEE International Conference on Systems, Man and Cybernetics.*

Marie, R. and A. Jean Marie (1993). Quantitative evaluation of discrete event systems: Models, performances and techniques. *Proc. of 5th IEEE International Workshop on Petri Nets and Performance Models* pp. 2–11.

Marsan, M.A., G. Balbo and G. Conte (1986). *Performance Models of Multiprocessor systems.* MIT Press. London.

Miyagi, P.E., E. Villani and N. Maruyama (2001). Modeling of hybrid supervisory systems using uml and petri nets. *Proc. of 8th International Conference on Emerging Technologies and factory automation* **1**, 149–158.

Moscato, F., M. Iacono and N. Mazzocca (2003). Coupling formal methods in a performance modelling methodology for heterogeneous supervisory systems. *Proc. of European Simulation and Modelling Conference.*

Murata, T. (1989). Petri nets: Properties, analisys and applications. *Proceedings of the IEEE* **77**(4), 541–580.

Vittorini, V., M. Iacono, N.Mazzocca and G. Franceschinis (2003). Osmosys: a new approach to multi-formalism modeling of systems. *Journal of Software and System Modeling. Accepted for publication.*

HIGH PERFORMANCE COMPUTING SIMULATION

COMBINING GLOBUS AND MOSIX FOR MPI CALCULATIONS

Adam Kozakiewicz,
Andrzej Karbowski
Institute of Control and Computation Engineering,
Warsaw University of Technology,
Nowowiejska 15/19,
00-665 Warsaw,
Poland,
E-mail: akozakie@ia.pw.edu.pl

Krzysztof Malinowski
NASK (Research and Academic Computer Network),
Wąwozowa 18,
02-796 Warsaw,
Poland

KEYWORDS

Grid infrastructure, performance tests, distributed computation, eigenvectors.

ABSTRACT

This paper presents a simple architecture for a campus-wide grid utilizing the resources of computer laboratories, designed for MPI calculations. The environment combines several tools, including the Globus Toolkit and MOSIX. A small experimental grid was set up and tested on a real-life example – finding an eigenvector of a large sparse matrix (Google's PageRank algorithm).

INTRODUCTION

Access to HPC resources is a must for today's scientists. The problems grow, algorithms are developed and any amount of computing power can be quickly consumed. Unfortunately, a real HPC center is very expensive. For many applications a relatively small cluster of PCs would be enough, but even setting up such a cluster requires some resources. At the same time a lot of processing power is lost in computer laboratories, left unused for many hours a day.

The goal of our proposal is to build a grid (see e.g. [Foster and Kesselman 2003]) integrating those resources, with the following characteristics:

- Cheap construction – no expensive commercial software,
- Ease of use – using the grid should require as little knowlegde of the underlying software as possible,
- Dynamic reconfiguration – it should be possible to detach any of the computers for some time without affecting the whole grid. Newly connected machines should be able to share the load quickly.

In this paper we show how to build such a grid by combining several available software tools into a single system, which, although complicated in structure, can be used by researchers with reasonable experience almost without preparation or learning new commands. As our experience has shown, the system is quite effective as a simulation environment. Section "The Grid" deals with the details, including a description of a small proof-of-concept test grid. The grid was tested using a real-life example – an implementation of the Google PageRank algorithm. The problem and test results are described in section "The Problem". We finish the paper with some general notes and observations in section "Conclusions".

THE GRID

The goals named in section "Introduction" can be met using two different approaches:

- A unified, scheduling grid, using specialized software. Requires learning new tools, but isn't very complicated. Each machine is a separate resource connected to the grid – a grid of machines. This approach is used by Condor (see e.g. [Thain 2003]).
- A hierarchical structure, treating each computer room as a single block connected to the grid, with variable resources – a grid of clusters. A combination of software tools on different levels.

We will focus on the second approach. Although seemingly unnecessarily complicated, it allows the construction of a very clear and easy to use setup.

System Architecture and the Choice of Tools

The system has two separate layers – the grid layer, connecting computer rooms to create a single virtual machine, and the cluster layer, connecting the machines in one room to offer their resources in the grid as a single virtual machine. The two layers are completely different, with different problems and different software solutions.

Both layers are unified from the end users point of view. The end user doesn't need to know the details of the

implementation, as for him the grid is simply an MPI-based cluster – MPI (Message Passing Interface) is a well-known standard for distributed programming.

The Cluster Layer

In this layer we deal with a set of machines under control of one administrator. The machines need to be detached from the grid during the courses and reattached afterwards. Ideally the process should not involve the administrator, it should be simple enough to be carried out by the students or laboratory staff leaving the room. The machines cannot be part of the grid when students work with them, since the machine could be incorrectly rebooted, killing the tasks being processed. Making individual machines visible from the MPI level creates a problem, as they can be added or removed during computation, requiring process migration – a difficult task.

The suggested solution uses a MOSIX ([Barak and La'adan 1998], [Barak et al. 1999], downloadable from the MOSIX Project homepage, http://www.mosix.com) cluster and a dedicated machine in each room. Only the one machine is visible as part of the grid, and it is the only one with MPI installed. All processes are running on this machine. When other machines are connected during computation, they share the load as MOSIX migrates part of the processes to them. From the grid level the cluster is a single machine with variable computing power. The cluster can also be used separately from the grid – tasks can simply be run from the access station, or, as a matter of fact, from any station.

It has to be noted, that MOSIX on Linux can't migrate processes using shared memory. Therefore, during instalation it is important to make sure that the MPI installed will not use shared memory for communication between processes running on the same platform. It is, for example, not easily configurable in LAM MPI 7.x (an option has to be given to mpirun, the default choice is hardcoded). In any other case the processes migrating away will communicate with the others as if all were running on the same computer.

For computers working under Linux by default the configuration is trivial and (dis)attaching them from the cluster is done simply by starting or stopping the mosix service. Computers running other operating systems have to be rebooted to Linux to join the cluster. As the stations, except the dedicated access point, don't need much except the kernel, a simple bootable floppy or CD seems to be the best choice.

The Grid Layer

In this layer we have to connect several rooms with different security policies, different administration, account names, even for the same users, different passwords. The connection may cross firewalls, and so on. This complicated situation is typical for a true grid and requires grid software.

We chose the Globus Toolkit 2.4 (see the Globus Project homepage at http://www.globus.org as it is rather popular, well tested and well integrated with MPI (MPICH-G2, downloadable from http://www.hpclab.niu.edu/mpi, see [Karonis et al. 2003]). Only the Resource Management pillar of the Toolkit is necessary for passing messages and starting remote jobs. The Globus Toolkit provides uniform login using certificates, simplifying the construction of the grid.

The communication between processes is provided by the MPICH-G2 package – an implementation of MPI 1.1 using Globus as the communication device. Unfortunately, this type of communication is not very efficient. The solution to that would be using an mpi flavor of Globus, with local communication done in a simple way. Unfortunately, we were unable to do that – use of MPICH variants is discouraged, MPICH-G2 could not be installed on top of LAM and we didn't have any other implementations available. Even so, the cluster was still rather effective, as the test results will show.

Test Grid

For testing we have used 9 computers from our small MOSIX cluster (a total of 11 × Pentium III/450MHz/128MB, running RedHat Linux 7.3, kernel 2.4.22 + MOSIX 1.10.1). We have reconfigured those machines into two separate clusters – 4 and 5 machines. The setup is shown in Figure 1.

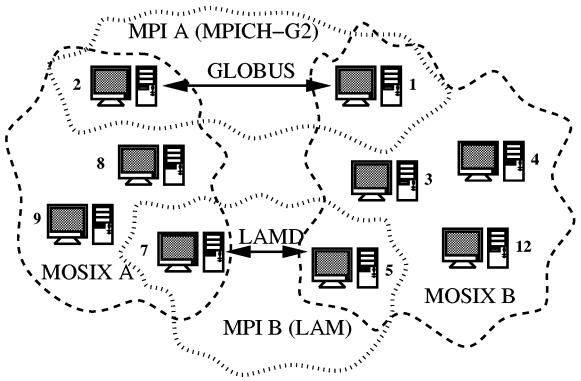


Figure 1: The Setup of the Test Grid

On one machine from each group we installed the Globus Toolkit 2.4 – resource management client, server and SDK. On top of it we built MPICH-G2, setting it to only connect the two machines. This configuration allowed us to test running programs on one, two machines or two clusters. To compare the performance with a more conventional MPI setup we also set up LAM MPI on one computer from each group, different

than MPICH-G2. The clusters MOSIX A and MOSIX B are separate. Only standard migration takes place, all messaging is done as if all processes still were on the same computer. The MPI A section is the test "grid". Job execution is done using the Globus Toolkit. The section MPI B is used for comparison only, and runs LAM MPI. MPI A and MPI B are never used together.

The tests made clear an important configuration element. The Globus Gatekeeper, the daemon responsible for running remote jobs, is started by the xinetd daemon. Under MOSIX xinetd, and most other daemons, is "locked", i.e. it cannot migrate. This state is inherited by child processes, so without changes the processes would never leave the cluster MPI A. The first solution was to have the programs unlock themselves. It helped, but the stations in cluster MPI A, when the task was too decomposed (18 processes) still were overwhelmed by processes spawned by the Globus tools. The solution is to configure the gatekeeper service as migratable (running it via mosrun -l) – not only can the programs skip unlocking themselves in this case, but the communication processes will migrate too.

THE PROBLEM

For testing we have chosen a large scale linear algebra problem. We implemented an algorithm of ranking WWW pages – Google PageRank – which is in fact a calculation of an eigenvector using the power method (see [Page et al. 1998], [Arasu et al. 2002]).

The Google PageRank Algorithm

The PageRank algorithm attempts to rank pages based on gathered data, so that the most often referred to pages get the highest ranking. Additionally it attempts to give more weight to links from pages with few links.

The links are represented by a binary $n \times n$ matrix G, usually huge and very sparse. Each page is characterised by two values: in-degree $c_j = \sum_i g_{i,j}$ and out-degree $r_i = \sum_j g_{i,j}$.

The model describes a motion of a random user, following with probability p random links from each page he visits, and with probability $1 - p$ jumping to a different address, ignoring existing links.

The matrix A, defined as follows:

$$a_{i,j} = pg_{i,j}/c_j + (1 - p)/n$$

is a probability matrix of a Markov chain. The sums of all columns are equal to 1. According to the Perron-Frobenius theorem the largest eigenvalue of this matrix is equal to one. The respective eigenvector is the solution to the equation $x = Ax$. The elements of this vector, normalized so that $\sum_i x_i = 1$, are the steady-state probability distribution of the Markov chain and are interpreted as PageRank of respective pages.

The simplest (and scalable) algorithm of solving this problem is the iteration:

$$x_{k+1} = Ax_k.$$

The algorithm is partially asynchronously convergent, see e.g. [Bertsekas and Tsitsiklis 1989].

Implementation

The algorithm has been implemented in a simple way, without excessive optimization. The matrix A is generated by a separate program and always stored in a sparse form – it can easily be too big to fit in memory. The main program is written using MPI, according to the SPMD rule. The matrix A is split between processes, each computes a different subvector of x. To minimize communication penalties and to utilize the asynchronous properties of the algorithm, the multiplication of a subvector in each process is repeated until a local solution is found, only then messages with results are exchanged and the next iteration begins.

To get the maximum gain from MOSIX process migration, our program declares itself as a cpu-intensive task. Otherwise the statistics gathered in the first stage, during the loading of data from disk, would suggest an I/O-intensive job, reluctantly migrated, and processes would collide on their home machine for several seconds, until those statistics decay.

Test Results

As a test we run the PageRank algorithm for 25000 pages and for 1000 pages. We ran several passes, in different configurations, with the same data. The larger task was well suited for distribution, the smaller one was too short – a stress test of the MPI's remote job management and communication procedures.

On a single machine and without decomposition (no communication, local and global convergence are the same) the larger task was done in 8 minutes The time can be further reduced to 6 minutes and 8 seconds by removing all MPI-related code and the final iteration, verifying convergence, unnecessary in this case. Decomposition introduced the need for communication and multiple iterations, slowing the computation down to 17 minutes, more if more processes were created. The smaller task could be solved even under 2 seconds.

To make use of the network, we connected the two machines of MPI A (see Fig. 1), but without activating MOSIX on them. We ran two tests, one with 2 processes, 1 per machine, and the other with 9 processes, our planned decomposition level for full cluster. The results for 9 processes are good, but definitely worse than without decomposition.

Table 1: Times (Minutes:Seconds) Required to Solve an Example Problem on Different Configurations

Task size	25000 pages		1000 pages	
MPI Implementation	MPICH-G2	LAM	MPICH-G2	LAM
A single machine, 1 process	8:00	7:59	12.7	3.1
A single machine, 2 processes	17:00	15:17	14.2	5.5
A single machine, 9 processes	20:43	17:22	40.0	6.4
A single machine, 18 processes	26:54	16:39	1:22.6	8.0
2 machines via MPI, 2 processes	8:01	7:46	13.3	3.1
2 machines via MPI, 9 processes	11:45	9:37	28.8	4.0
2 machines via MPI, 18 processes	12:40	8:17	46.2	4.8
9 machines total, 2 via MPI, the rest share the load via MOSIX. 9 processes	4:42	2:14	32.0	3.9
Same as above, 18 processes	5:52	2:19	56.9	4.7

Finally, we activated the mosix clusters. The computation was done in under 5 minutes. The 9 machines not only ran the program much faster than one machine, but decomposition actually proved useful – the cluster was much faster than a single machine with optimized, non-distributed code. At the same time the cluster's total memory potentially allows solving of even larger tasks.

For reference, all tests were repeated using the MPI B cluster, with a different implementation of MPI. The results are better, because of more effective communication. However, this approach could not be used for building a campus-wide grid, because of lack of functions offered by the Globus Toolkit, like single logon. This is the main advantage of MPICH-G2–based system, tested as MPI A. The tests have shown, that running remote jobs and communication are a lot more expensive under Globus. All results are summarized in Table 1.

In other, non-measured tests we turned on and off individual machines in the cluster (of course except the two nodes of MPI A). As long as the system was closed properly, the calculation was not seriously hampered by this.

CONCLUSIONS

The approach presented in this paper seems to be working. The combination of MPI working on few physically isolated machines and MOSIX spreading the load to other stations, connected only when available, allows for very comfortable and fast computation and utilization of computing reserves.

Spreading tasks from one or two nodes to the rest of the cluster by using MOSIX process migration mechanisms is not optimal if only speed is important. In this case it would be better to use MPI to spread the tasks to all machines from the beginning, and only let MOSIX manage the loads if tasks differ in complexity. However, the difference for larger tasks with long processes is very small, and this setup is far more elastic – all machines in the cluster except the "access point", where MPI is installed, can be detached and reattached during computation, for example during courses.

Neither MOSIX nor MPI is a high reliability environment. If a computer crashes for any reason, or is simply turned off without closing the system correctly, the processes running on it will be lost. Even though the design prefers detaching computers from the grid before laboratory work, a danger still exists. Therefore, programs for the grid should be written for reliability and fault tolerance, if they are expected to run excessively long.

The use of Globus as an MPI device to connect the clusters is a good choice from the security standpoint and allows for better administration of resources, but the penalties are noticeable. When several clusters with different user names, policies and system setup are connected into a grid, the Globus Toolkit solves the problem, but using Globus-based MPI in a single cluster is a waste of resources. A comparable MPI implementation using rsh and simple communication protocols worked significantly faster. A good way of dealing with the slowness of Globus in our example would be to use an mpi flavor of Globus (using an implementation of mpi for local passing of messages). This would allow faster local (in-cluster) communication. Unfortunately we couldn't achieve such a setup due to a lack of a suitable MPI implementation (non-MPICH based and not LAM).

The grid built this way is also very easy to use. The user only needs to know that he should use the `grid-proxy-init` command to login and `grid-proxy-destroy` to logout. That, and familiarity with MPI, especially the MPICH implementation, suffices. The system behaves mostly like a simple cluster. Our test cluster has already been used for distributed simulation of aerodynamic systems (flatter, see early

results before introducing Globus in [Dul 2004]) and hydrological systems (flood control in water retention system, work in progress, not yet published). The scientists running those simulations were not involved in the preparation of the cluster, but didn't require any significant training to start using it. Note, that due to high costs of communication the system is only applicable to some types of distributed simulation – especially if the simulation is not really distributed, but just run in several instances for several sets of parameters.

The structure shown in this paper is easy to set up. It can be suggested, wherever a large MPI computation is necessary and several smaller clusters can be connected. Because MOSIX offers dynamic load balancing, such a setup handles tasks with wildly varying sizes remarkably well, better than typical MPI environments with static allocation of tasks to machines. The tasks cannot be migrated between clusters (over MOSIX), so the initial distribution is still important.

SUMMARY

We have shown, how Globus and MOSIX can be combined to create an elastic and efficient grid for MPI calculations. We have shown, that it can indeed speed up large calculations, while using it for smaller tasks is a mistake – the costs of spawning remote processes are too high.

ACKNOWLEDGEMENT

The research presented in this paper was partially supported by the Foundation for Polish Science.

REFERENCES

[Arasu et al. 2002] Arasu, A.; J. Novak; A. Tomkins; and J. Tomlin. 2002. "PageRank Computation and the Structure of the Web: Experiments and Algorithms". World Wide Web 2002 Conference. available at http://www2002.org/CDROM/poster/173.pdf

[Barak and La'adan 1998] Barak, A. and O. La'adan. 1998. "The MOSIX Multicomputer Operating System for High Performance Cluster Computing". *Journal of Future Generation Computer Systems*, Vol. 13, No. 4-5 (Mar), 361-372.

[Barak et al. 1999] Barak, A.; O. La'adan; and A. Shiloh. 1999. "Scalable Cluster Computing with MOSIX for LINUX", *Proceedings of the 5-th Annual Linux Expo*, Raleigh, May, 95-100.

[Bertsekas and Tsitsiklis 1989] Bertsekas, D.P. and J.N. Tsitsiklis. 1989. *Parallel and Distributed Computation: Numerical Methods*, PrenticeHall, Englewood Cliffs, NJ, 1989

[Dul 2004] Dul, F.A.. 2004. "Aeroelastic Stability Analysis on Local Cluster with MOSIX Scheduler". *Proceedings of the 7th National Conference on Evolutionary Computation and Global Optimization*, (Kazimierz Dolny, Poland, May), Warsaw University of Technology, 221-229.

[Foster and Kesselman 2003] Foster, I. and C. Kesselman. 2003. *The Grid 2: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann Publishers Inc., San Francisco.

[Karonis et al. 2003] Karonis, N.; B. Toonen; and I. Foster. 2003. "MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface". *Journal of Parallel and Distributed Computing (JPDC)*, Vol. 63, No. 5 (May), 551-563.

[Page et al. 1998] Page, L.; S. Brin; R. Motwani; and T. Winograd. 1998. "The PageRank Citation Ranking: Bringing Order to the Web". Stanford Digital Library Technologies Project, available at http://dbpubs.stanford.edu:8090/ pub/1999-66

[Thain 2003] Thain, D.; T. Tannenbaum; and M. Livny. 2003. "Condor and the Grid". In *Grid Computing: Making the Global Infrastructure a Reality*, Berman F.; Fox G.; and Hey T. (Eds.). John Wiley & Sons Ltd., Chichester. 299-335.

AUTHOR BIOGRAPHY

ADAM KOZAKIEWICZ obtained his M.Sc. degree at the Warsaw University of Technology in 2001 and is currently a Ph.D. student there. Student member of IEEE. Interests include nonlinear optimization, distributed computation, modelling of computer networks, pricing and traffic control mechanisms.

ANDRZEJ KARBOWSKI obtained his M.Sc. and Ph.D. degrees at the Warsaw University of Technology (WUT) in 1983 and 1990. With WUT since 1983, currently as Assistant Professor, since 2001 also a researcher at NASK. Co-author and editor of a book on parallel computing. Member of IEEE. Interests include large scale systems, distributed computations, optimal control and management in risk conditions, control and decision problems in computer networks.

KRZYSZTOF MALINOWSKI graduated from WUT in 1971, in 1989 obtained the title of professor of technical sciences. Author or co-author of three books and over one hundred thirty journal and conference papers. Ordinary professor at WUT since 1994 and research professor at NASK since 2001. Currently Head of Research Division at WUT and Research Director at NASK. Member of the Polish Academy of Sciences. Interests include control of complex systems, hierarchical methods, distributed and decentralized systems, pricing mechanisms and ommunication networks.

LOAD BALANCING FOR A PARALLEL JOINT DIGONALIZATION OF SYMMETRIC MATRICES

Ales Holobar, Milan Ojstersek and Damjan Zazula
University of Maribor
Faculty of Electrical Engineering and Computer Science
Smetanova 17
2000 Maribor, Slovenia
E-mail: ales.holobar@uni-mb.si

KEYWORDS
Joint diagonalization, parallel Jacobi, large scale computing, distributed algorithms, load balancing

ABSTRACT

This paper discusses a novel load balancing strategy for the parallel joint diagonalization of real symmetric or complex Hermitian matrices on a cluster of heterogeneous personal computers. The theoretical performance of the parallel joint diagonalization algorithm is studied and the need for dynamic load balancing in the presence of varying resource availability is outlined. Considering the leading role of the first processing unit in the parallel joint diagonalization algorithm the centralized load balancing strategy is introduced. Load balancing scheme consists of initial data distribution, workload monitoring, decision of balancing, and workload redistribution. The theoretical expectations are confirmed by experimental results measured on heterogeneous clusters of up to 50 personal computers.

INTRODUCTION

Joint diagonalization (JD) of several matrices consists of finding a unique transformation which simultaneously transforms the matrices to be as close to diagonal as possible. The problem has been studied extensively during the last few decades and has proven to provide a solution to a wide range of scientific and engineer's problems, especially to the statistical problem of common principal components (Flury 1984), independent component analysis (ICA) (Hyvarinen et al. 2001), and blind signal separation (BSS) (Cardoso and. Souloumiac 1993). Joint parameters estimation technique which can be reduced to the problem of joint diagonalization cover the area of biomedical signal processing, digital communication systems, image restoration and understanding, audio and speech processing, array processing, separation of artifacts, and even finding hidden factors in financial data (Hyvarinen et al. 2001).
Consequently, a large variety of numerical methods for joint diagonalization were developed utilizing Jacobi iterations (Cardoso and. Souloumiac 1993; Belouchrani et al. 1998), theoretic matrix log likelihood functions (Nedeljkovic and Quinn 1992), Gauss-Newton optimizations (Van der Veen, 2001; Joho and Rahbar 2002) and alternating-directions algorithm (Yeredor 2002). However, applying the signal separation techniques to the convolutive mixtures a large number of such matrices have to be processed in order to suppress the influence of artifacts (noise). Furthermore, the matrices that enter joint-diagonalization are generally very large, thus, high computational complexity of joint diagonalization becomes an important issue. Recently a novel algorithm for parallel joint diagonalization of real symmetric or complex Hermitian matrices was introduced (Holobar et al. 2003). The approach is based on the Jacobi diagonalization and runs on clusters of heterogeneous personal computers with an arbitrary number of processing units. However in (Holobar et al. 2003) only preliminary performance analysis on the cluster of 7 personal computers was outlined. No influence of the initial load balancing and no load rebalancing strategy were discussed. As a consequence, the introduced diagonalization algorithm was unsuitable for use in the dynamic environment with varying resource availability.
In this paper a load balancing strategy for the parallel joint diagonalization (Holobar et al. 2003) of a large number of real symmetric or complex Hermitian matrices of arbitrary size is introduced. The dynamic load balancing scheme is based on centralized architecture and consists of initial data distribution, workload monitoring, decision of balancing, and workload redistribution. It minimizes the communication costs and proves to achieve a stable speedup factor and a good load balance for all possible input matrices and number of processors used. The paper is organized in 5 Sections. Section 2 briefly reviews the parallel Jacobi's algorithm for joint diagonalization of symmetric (Hermitian) matrices. Section 3 discusses its theoretical performance on the heterogeneous cluster of processing units. A novel load balancing strategy is introduced in Section 4, while Section 5 reveals its experimental performance on different number of personal computers. We conclude the paper with the discussion in Section 5.

PARALLEL JOINT DIAGONALIZATION

Consider a set of K Hermitian $N \times N$ matrices $\mathbf{A}_k$

$$\Theta = \{\mathbf{A}_k; k = 1,...,K\} \quad (1)$$

We search for such a unitary matrix $\mathbf{U}$ which diagonalizes given set of matrices $\mathbf{A}_k$:

$$\mathbf{D}_k = \mathbf{U}^H \mathbf{A}_k \mathbf{U} \quad (2)$$

where $\mathbf{D}_k$ is a diagonal N×N matrix and ^{H} denotes the complex conjugate transpose. The idea behind Jacobi

method is to use successive orthogonal transformations **J** in order to reduce the norm of the off-diagonal elements (Golub and Van Loan 1996)

$$off(\mathbf{\Theta}) = \sum_{k=1}^{K} \sqrt{\sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} (a_{ij}^{k})^2} \qquad (3)$$

where a_{ij}^{k} denotes the (i,j)-th element of the matrix $\mathbf{A}_k$. Transformations J are defined as Jacobi (Givens) rotations of the form (Golub and Van Loan 1996)

$$\mathbf{J}(p,q,c,s) = \begin{bmatrix} 1 & \cdots & 0 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & & \vdots & & \vdots \\ 0 & \cdots & c & \cdots & s & \cdots & 0 \\ \vdots & & \vdots & \ddots & \vdots & & \vdots \\ 0 & \cdots & -s & \cdots & \bar{c} & \cdots & 0 \\ \vdots & & \vdots & & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & 0 & \cdots & 1 \end{bmatrix} \begin{matrix} \\ \\ p \\ \\ q \\ \\ \\ \end{matrix} \qquad (4)$$
$$\qquad\qquad\qquad\qquad\quad p \qquad\quad q$$

where p and q stand for indices of rows and columns, respectively, *(c,s)* is complex rotation cosine-sine pair with $|c|^2 + |s|^2 = 1$ and $\bar{c}$ denotes the conjugate value of c. Jacobi rotations are clearly orthonormal.

Setting the matrix **U** equal to the $N \times N$ identity matrix ($\mathbf{U}=\mathbf{I}_{N\times N}$) the basic step of joint Jacobi method involves (Golub and Van Loan 1996):

1. choosing an index pair *(p,q)* which satisfies $1 \le p < q \le N$,
2. computing a cosine-sine pair *(c,s)*
3. replacing $\mathbf{A}_k$ with $\mathbf{A}_k = \mathbf{J}^T \mathbf{A}_k \mathbf{J}$ where $\mathbf{J}=\mathbf{J}(p,q,c,s)$,
4. replacing **U** with $\mathbf{U}=\mathbf{J}^T\mathbf{U}$,

where the cosine-sine pair *(c,s)* in each Jacobi step is calculated as follows: for a given pair of indices *(p,q)* define a *3×3* matrix **G** as (Cardoso and Souloumiac 1996)

$$\mathbf{G}(p,q) = real\left(\sum_{k=1}^{K} \mathbf{h}^{H}(\mathbf{A}_k, p, q)\mathbf{h}(\mathbf{A}_k, p, q) \right), \qquad (5)$$

where

$$\mathbf{h}(\mathbf{A}_k, p, q) = \left[a_{pp}^{k} - a_{qq}^{k}, a_{pq}^{k} + a_{qp}^{k}, i(a_{qp}^{k} - a_{pq}^{k}) \right], \qquad (6)$$

and a_{pq}^{k} denotes the *(p,q)*-th element of matrix $\mathbf{A}_k$. Then, under the constraint of $|c|^2 + |s|^2 = 1$, Eq. (3) is minimized if

$$c = \sqrt{\frac{x+r}{2r}}, \quad s = \frac{y}{\sqrt{2r(x+r)}}, \quad r = \sqrt{x^2 + y^2} \qquad (7)$$

where $[x,y]^T$ is any eigenvector associated with the largest eigenvalue of **G** (Cardoso and Souloumiac 1996).

Each Jacobi transformation $\mathbf{J}(p,q,c,s)$ changes only the p-th and q-th rows and columns of matrices $\mathbf{A}_k$, respectively. To guarantee the convergence of Jacobi method, all possible pairs of *(p,q)* should be generated. By choosing pairs *(p,q)* wisely, *N/2* Jacobi rotations can be simultaneously applied (Holobar et al. 2003). In the sequel, the cycle over all possible *(p,q)* pairs, i.e. *N(N-1)/2* Jacobi rotations reducing all non-diagonal elements, will be referred to as a sweep (Golub and Van Loan 1996).

The basic idea behind the parallel joint diagonalization (Holobar et al. 2003) is to distribute the matrices $\mathbf{A}_k$ among the different processing units (PU) and let each PU diagonalize its own subset of matrices. To guarantee the generation of universal transformation matrix **U** which is common to all matrices $\mathbf{A}_k$ the *3×3* matrices $\mathbf{G}(p,q)$ must now be **c**onstructed globally, i.e. summed up over all PUs (Holobar et al. 2003)

$$\begin{aligned} \mathbf{G}(p,q) &= real\left(\sum_{k=1}^{R_1} \mathbf{h}^{H}(\mathbf{A}_k, p, q)\mathbf{h}(\mathbf{A}_k, p, q) \right) + \\ &+ real\left(\sum_{k=1}^{R_2} \mathbf{h}^{H}(\mathbf{A}_k, p, q)\mathbf{h}(\mathbf{A}_k, p, q) \right) + \ldots + \\ &+ real\left(\sum_{k=1}^{R_Q} \mathbf{h}^{H}(\mathbf{A}_k, p, q)\mathbf{h}(\mathbf{A}_k, p, q) \right) = \\ &= \mathbf{G}_1(p,q) + \mathbf{G}_2(p,q) + \ldots + \mathbf{G}_Q(p,q) \end{aligned} \qquad (8)$$

where Q stands for the number of PUs and each R_i; $i=1,\ldots,Q$ with $R_1+R_2+\ldots+R_Q=K$ denotes the number of matrices being joint diagonalized by the i-th PU. Summing up the $\mathbf{h}(\mathbf{A}_k,p,q)$ vectors over the corresponding matrix subset, each PU calculates its local matrix $\mathbf{G}_i(p,q)$. The latter are then globally added (Holobar et al. 2003) to from the entire matrix $\mathbf{G}(p,q)$ as defined in (8).

Recall $\mathbf{G}(p,q)$ is a *3×3* matrix and its eigendecomposition is a trivial task. Therefore, the computation of the cosine-sine *(c,s)* pair can be performed by a single PU. Afterwards, *(c,s)* pair is propagated back to all PUs where the local copies of the rotation matrix $\mathbf{J}(p,q,c,s)$ are constructed (Fig. 1). Pseudocode of parallel Jacobi joint diagonalization (Holobar et al. 2003) is outlined in Fig. 1.

```
until( the convergence condition is met )
    For all N/2 index pairs (p,q) calculate the local Gi(p,q) matrices.
    if (PU_ID=1) // the first PU
        For each pair of indices (p,q) wait to receive all the local matrices Gi(p,q) and add them to form the global matrix G(p,q).
        For each pair of indices (p,q) form the rotation matrix J(p,q,c,s) and update the matrix U: U=J(p,q,c,s)U.
        Send the cosine-sine pairs (c,s) to all the other PUs
    else
        Send the local matrices Gi(p,q) to the first PU.
        Wait to receive the cosine-sine pair (c,s) from the first PU and form the rotation matrix J(p,q,c,s)
    endIf
    For all N/2 index pairs (p,q) update the matrices Ak:
    Ak=J(p,q,c,s)^H Ak J(p,q,c,s)
    Form new (p,q) pairs by shifting the indices according to the tournament ordering.
endUntil
```

Figure 1: Pseudocode of Parallel Jacobi Joint Diagonalization

THEORETICAL PERFORMANCE ANALYSIS

In this section, we compare the theoretical performances of the parallel and sequential Jacobi joint diagonalizations. They both share the same number of required sweeps. Unfortunately, no general proof of their global convergence rate can be made (there may even be no similarity transformation which simultaneously diagonalizes the whole set of matrices $\mathbf{A}_k$). In the sequel, our discussion will be limited to the level of one sweep while the basic floating point operation will be referred to as a flop.

Communication complexity

Global addition of local *3×3* $\mathbf{G}_i(p,q)$ matrices in (8) represents minor network load. Assuming *4* bytes representation of floating point data type, the total amount of *36Q* bytes per each global addition, i.e. each Jacobi rotation, is transferred over the network, where *Q* denotes the number of PUs. Dispatching of the cosine-sine pairs *(c,s)* requires additional *8Q* bytes per rotation. Having *N(N-1)/2* rotations per sweep the total communication costs yield:

$$C_{\text{parallel}}^{sweep} = \sum_{i=1}^{Q}\left[23N(N-1)\varepsilon_{1i} + (N-1)\sigma_{1i}\right] \qquad (9)$$

where ε_{1i} denotes the time required to transfer 1 byte from the first PU to the i-th PU and σ_{1i} denotes the start-up time to initiate the communication activity between the first and i-th PU (typically in the order of tens of microseconds). Note the amount of the transferred data is independent of the number of matrices $\mathbf{A}_k$, but increases with their size and the number of PUs.

Computational complexity

The main computational cost in sequential Jacobi procedure is the update of the matrices involved in the diagonalization. In each Jacobi step, we compute the local matrix $\mathbf{G}_i(p,q)$ which requires *12K* flops and update the matrices $\mathbf{A}_k$ which takes additional *12KN* flops. Performing *N(N-1)/2* rotations per sweep, the computational complexity of sequential Jacobi procedure on the i-th PU yields:

$$T_{sequential}^{sweep}(N,K) = \gamma_i\left[6KN(N-1)(N+1) + \frac{N(N-1)}{2}\beta\right] \qquad (10)$$

where γ_i stands for the time required to execute one flop on the i-th PU and β denotes the computational complexity (in flops) of *3×3* eigendecomposition. The exact computational complexity of the eigendecomposition depends on the numerical library used, hence, general denotation is used in this paper.

Running a parallel version of Jacobi joint diagonalization additional waiting time arises w_i as a consequence of required global synchronization. After each step of a sweep (sweep step comprises *N/2* Jacobi rotations) the first PU must wait to receive the local matrices $\mathbf{G}_i(p,q)$ from all the PUs (all-to-one communication) and, afterwards, it dispatches the calculated cosine-sine pairs *(c,s)* to all other PUs (one-to-all communication). The computational costs per sweep on the i-th PUs reach

$$T_i^{sweep}(N,K) = \gamma_i\left[6R_iN(N-1)(N+1)\right] + (N-1)w_i \qquad (11)$$

where R_i denotes the number of matrices being diagonalized on the i-th PU. Waiting times caused by the differences in processing capabilities of different PUs can be approximated by

$$w_i \approx \frac{N}{2}\beta\gamma_i + \max_j(\alpha_j) - \alpha_i \qquad (12)$$

with a computation time difference

$$\alpha_i = \left|R_i\gamma_i - R_1\gamma_1\right|\left[6N + 6N^2\right]. \qquad (13)$$

Comparing the sequential and parallel algorithms, the achieved speedup factor (a ratio between the total processing time of sequential and parallel Jacobi algorithm) yields

$$\frac{T_{parallel}^{sweep}}{T_{sequential}^{sweep}} \approx \frac{1}{F_1^{optimal}} + \frac{\max_i(R_i\gamma_i - R_1\gamma_1)}{K} \qquad (14)$$

where $F_1^{optimal}$ denotes the optimal speedup factor (a ratio between the joint computational power of all PUs and the computational power of the first PU).

Eq. (14) clearly demonstrates how the computation time differences α_i degrade the performance of the parallel joint diagonalization algorithm. It is, therefore, essential to keep the computation time differences α_i as small as possible.

LOAD BALANCING

Running the presented algorithm on heterogeneous cluster of personal computers with PUs with different processing speed and different amount of memory, it is essential to achieve a good load balancing for all possible configurations of input matrices and number of PUs used. A smart adaptive load balancing minimizes the computation time differences α_i and, consequently, the required waiting times w_i. Moreover, computation and communication capabilities of each PU may vary in time due to the other applications competing for resources. To maximize the system utilization load balancing should also be dynamic. Finally, considering the leading role of the first PU in our algorithm the centralized load balancing architecture is most appropriate; the first PU acts as load balancer (master) while all the others PUs serve as slaves.

In our load balancing scheme every PU runs a statistics about the amount of time spent at synchronization points. When sending the local $\mathbf{G}_i(p,q)$ matrices the slaves also send the performance information to the master, which then calculates the aggregate computation capability of the entire system and generates the instructions to rebalance the load (redistribute the matrices). In order to reduce the amount of transferred data, the matrices are shifted directly between the slaves.

There are many parameters influencing the stability and efficiency of work movement. Among them the frequency of load balancing is most important (Siegell and Steenkiste 1997). The rebalancing of the PUs' workload takes $2Q$ bytes per Jacobi rotation for dispatching the performance information of each slave and $4N^2$ bytes per transferred matrix. Computational costs per sweep now yield

$$C_{parallel}^{sweep} = \sum_{i=1}^{Q}\left[23N(N-1)\varepsilon_{ij} + 4N^2 S_{ij}\varepsilon_{ij} + (N-1)\sigma_i + S_{ij}\sigma_i\right] \quad (15)$$

where S_{ij} stands for the number of matrices that are transferred between the i-th and the j-th PU due to the load balancing.

The rebalancing costs are relatively high and can easily outweigh the benefits. Quick and small changes in load must, hence, be ignored and we must be sure that the new distribution of matrices will improve the system performance. In our solution the load balancer (the first PU) keeps the history of the number of matrices being processed on each PUs and their waiting times. Knowing the exact period between the two synchronization points and the waiting time on each node, the master calculates the PUs' utilizations and sets their imbalance values:

$$Tr_{load}(i) = \frac{waiting_time(i)}{processing_time(i)} \quad (16)$$

All the PUs whose imbalance value exceeds the predefined treshold (typically set between 1 and 3) enter the negotiations for additional matrices. Although redistribution of matrices can be performed after each sweep step (before new *(p,q)* pairs are formed), the load balancer decides to rebalance the load only after the imbalance threshold has been exceeded by the same PU in several (typically 50) consequential sweep steps. The matrices are taken from the slowest PU (the PUs with smallest value of imbalance threshold) and are proportionally distributed among the PUs exceeding the predefined threshold.

EXPERIMENTAL REULTS

The presented parallel joint diagonalization algorithm was implemented in Microsoft Visual C++. The communication among different PUs was carried out using the MPICH implementation (Argonne National Laboratory 2003) of the Message Passing Interface standard. All the tests were conducted on heterogeneous clusters of 1, 10, 20, 30, 40, and 50 personal computers. All cluster nodes were running Windows XP operating system and were connected with 100 Mbps network. The technical details about the PUs constituting the clusters are presented in Table 1. Due to its computational superiority and large amount of memory the role of the master (load balancer) was always assigned to the first PU. The nodes constituting the cluster were located in 5 different classrooms at the Faculty of Electrical Engineering and Computer Science, University of Maribor, Slovenia, while all communication packages were routed via standard network (no special network connections were utilized).

Table 1: Declared Characteristics of PUs Constituting the Test Clusters of 1 to 50 Personal Computers

PU ID	Total memory	CPU, declared speed
1	1.5 GB	Intel Pentium 4, 2.00 GHz
2	500 MB	Intel Pentium 4, 1.80 GHz
3-15	256 MB	Intel Pentium 4, 1.60 GHz
16-26	256 MB	AMD Athlon 2000+, 1.667 GHz
27-39	256 MB	Intel Celeron 4, 1.70 GHz
40	512 MB	AMD Athlon 2600+, 2.09 GHz
41-50	256 MB	Intel Celeron 4, 1.70 GHz

As a first step in all simulation runs the initial load distribution was calculated. Firstly, the computing power of each node was determined experimentally by measuring the time spent to multiply 1000 random matrices of size *100×100* on each node. Secondly, matrices were distributed proportionally to the estimated computing power of each node. The imbalance threshold Tr_{load} was set to 3 while the load balancing was only initiated if the same slave exceeded the threshold in 50 successive sweep steps.

We begin with comparison of the time required to diagonalize the same matrices on a different number of PUs. Groups of *200*, *400*, *600* and *800* random symmetric matrices of size *400×400* were generated. The total execution time and time spent for communication between the PUs were averaged over 10 simulation runs. The speed-up factor achieved by the parallel implementation of the joint diagonalization in dependence of the number of PUs is depicted in Fig. 2. For the sake of clarity, the computational powers of PUs constituting the cluster were equalized and proportionally scaled processing times are displayed. Note also the times required to diagonalize different number of matrices on a single PU are all normalized to 1.

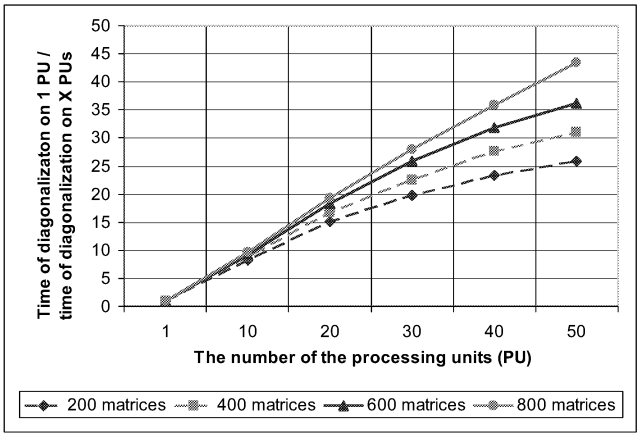


Figure 2: Total Diagonalization Time Speed Up in Dependence of the Number of Processing Units when Joint Diagonalizing Different Numbers of 400×400 Matrices

In the next experiment, the performance of parallel joint diagonalization was measured against the size of matrices. Replacing the number of matrices with their size, the same protocol was followed as in the first experiment. Groups of *400* symmetric matrices of size *100×100*, *200×200*, *400×400* and *600×600* were first randomly generated. The total

execution time and time spent for communication between the PUs were averaged over 10 simulation runs, while, as in the first experiment, new random matrices were generated in each run. The results are depicted in Fig. 3 (they should be interpreted in the same way as Figs. 2).

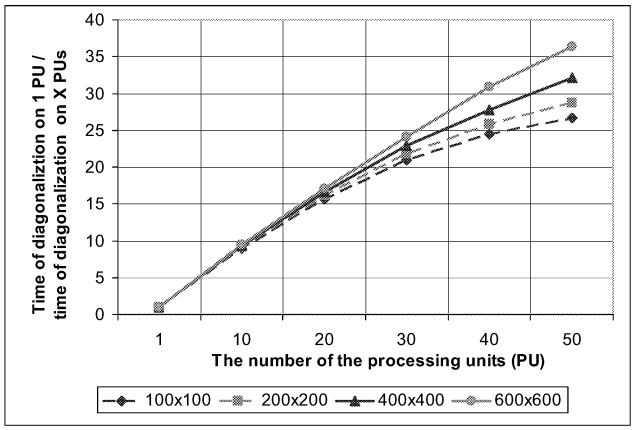


Figure 3: Total Diagonalization Time Speed Up in Dependence of the Number of Processing Units when Joint Diagonalizing 400 Matrices of Different Size

CONCLUSION

The presented load balancing strategy for parallel joint diagonalization proved to be very efficient and highly scalable and adaptable to an arbitrary number of PUs. Communications between PUs (including the matrices which were transferred due to the load balancing) have proved to be a minor cost, especially when a large number of matrices is diagonalized. Diagonalization of small number of large matrices proved to be less efficient. The reason can be found in the initial load balancing. Namely, limiting the basic load distribution unit to a single matrix, the load balancing is most efficient when processing a large number of relatively small matrices. On the other hand, processing a small number of large matrices induces large differences in amount of time spent on each node while waiting for global synchronization. As result, the communication costs rise and the system performance drops, as demonstrated by the experimental results (Figs. 2 and 3).

On the other hand, both theoretical and experimental results prove the relative contribution of communication activity to the overall costs is proportional to the number of nodes but decreases linearly with both the number and the size of the matrices. Diagonalizing 800 matrices of size *400×400* on 50 PUs, the speed-up factor yields *43.5* and improves with the number of matrices (Fig. 2). Diagonalizing 400 matrices of size *600×600* on 50 PUs a speed-up ratio drops to 36.3 (Fig. 3) but increases with the size of matrices.

Evidently, our load balancing strategy relies on the assumption that previous waiting times are indicative of future computational capabilities. This assumption is certainly questionable in highly competitive environments.

ACKNOWLEDGEMENT

This work was supported by the Slovenian Ministry of Education, Science, and Sport (Contract No. S2-796-010/21301/2000), and by the European funding in the 5th Framework project entitled NEW (Contract No. QLK6-2000-00139).

REFERENCES

Argonne National Laboratory. 2003 *MPICH - A Portable Implementation of MPI,* http://www-unix.mcs.anl.gov/mpi/mpich/indexold.html

Belouchrani, A., K.A. Meraim. 1998. "Blind Source Separation based on time-frequency signal representation", *IEEE trans. On Signal Processing*, Vol. 46, 2888-2898.

Cardoso, J. F. and A. Souloumiac. 1993. "Blind Beamforming for Non-Gaussian Signals", *IEE Proceedings-F 6* 362–370.

Cardoso, J. F. and A. Souloumiac. 1996. "Jacobi angles for simultaneous diagonalization", *SIAM J. Mat. Anal. Appl.*, Vol. 17, 161-164.

Flury, B. N. 1984. "Common principal components in K groups", *J. Amer. Statist. Assoc.*, Vol. 79, 892-897.

Golub, G. H. and C. F. Van Loan. 1996. *Matrix computation*, 3rd ed., The Johns Hopkins University Press, Baltimore.

Holobar, A., M. Ojstersek and D. Zazula. 2003. "A new approach to parallel joint diagonalization of symmetric matrices.", In *Proc. of EUROCON 2003 Conference* (Ljubljana, Slovenia) IEEE, Picataway, Vol 2, 16-20.

Hyvarinen, A., J. Karhunen and E. Oja. 2001. *Independent Component Analysis*, Hohn Wiley & sons, Inc. New York.

Joho, M. and K. Rahbar. 2002. "Joint diagonalization of correlation matrices by using Newton methods with application to blind signal separation" In *Proc. SAM* (Rosslyn, VA) 403-407.

Pham, D. T. 2001. "Joint approximate diagonalization of positive definite Hermitian matrices", *SIAM J. Matrix Anal. and Appl.* Vol. 22, 1136-1152.

Siegell, B. and P. Steenkiste. 1997. "Automatic Selection of Load Balancing Parameters Using Compile-time and Run-time Information", *Concurrency: Practice and Experience*, Vol. 9, 275-317.

Van der Veen, A. J. 2001 "Joint diagonalization via subspace fitting techniques", In *Proc. ICASSP Vol. 5* (Salt Lake City, UT) 2773–2776.

Yeredor, A. 2002. "Non-orthogonal joint diagonalization in the least squares sense with application in blind source separation", *IEEE Trans. Signal Processing*, Vol. 50, 1545–1553.

ALES HOLOBAR received his B.S degree in computer science from the University of Maribor, Maribor, Slovenia, in 2000. He is currently a PhD student at the Faculty of Electrical Engineering and Computer Science, University of Maribor, Slovenia. In 1997, he joined the System Software Laboratory at Faculty of Electrical Engineering and Computer Science, Maribor, Slovenia, where he works as a researcher.

His research interests are in the areas of virtual reality, distributed systems and signal processing, with current activity focused on parallel processing and blind source separation.

A WEB BASED APPROACH TO DISTRIBUTE SIMULATION-BASED FAULT INJECTION CAMPAIGNS

Miguel Mateo, David de Andrés, Juan Carlos Baraza, Lenin Lemus, Pedro Gil
Departamento de Informática de Sistemas y Computadores (DISCA)
Universidad Politécnica de Valencia
Camino Vera s/n
46022, Spain
E-mail:{mimateo, ddandres, jcbaraza, lemus, pgil}@disca.upv.es

KEYWORDS

Distributed simulation, fault injection tools, web services.

ABSTRACT

Simulation-based fault injection is a hard and time-consuming task that usually involves multiple executions of the same model. This paper presents preliminary experiences with DVFIT-w, a web-based distributed version of VFIT, a fault injection tool for VHDL models. The main objective of DVFIT-w is to reduce the time needed to perform a fault injection campaign. Results show that this objective has been achieved and the obtained speed-up scales well when increasing the number of concurrent servers. Moreover, the new tool presents new capabilities such as storing the results in web servers for later retrieval or storing models for new executions. These new capabilities increase the utility and flexibility of the tool compared with the previous sequential version. In order to make easier the development of more complex web-oriented distributed applications, the tool is based on well-known technologies like HTTP, SOAP and TCL.

INTRODUCTION

Validating the dependability of systems is a challenging task, but it is necessary in the design of Fault Tolerant Systems (FTS). The dependability validation can be carried out in two different ways: a) by observing the behaviour of the system during its operative phase and waiting for faults to appear, or b) by means of controlled experiences, i.e., introducing faults deliberately. This second method is called fault injection. Fault injection is more suitable to validate FTSs because the time needed to study the system is shorter than observing the real operative system.

The three main fault injection techniques are: *physical fault injection* (or Hardware Implemented Fault Injection, HWIFI), *Software Implemented Fault Injection* (SWIFI) and *Simulation-Based Fault Injection*. In *Simulation-Based Fault Injection*, a model of the system is simulated in a computer system; therefore, it can be used in early system design phases, because an implementation of the system is not needed. We are interested in this kind of fault injection techniques and systems.

Our workgroup has developed a simulation-based fault injection tool for VHDL models: VFIT (VHDL Fault Injection Tool) (Gil et al. 2003). This tool runs on a PC (or compatible) under Microsoft Windows™ operating systems, using a commercial simulator for the execution of simulations (ModelSim©, by Model Technology).

VFIT was designed as a simple and easy-to-use model independent tool, suitable for defining and executing fault injection campaigns into systems of any complexity. A fault injection campaign consists of the different fault injection simulations to execute for validating the dependability of a system model.

VFIT performs the fault injection simulation process in four phases (Figure 1):

1. Instrumentation and model checking. VFIT analyses the model and builds up some internal data structures needed for next phases.
2. Campaign parameters specification. The user defines the number of simulations, the injection targets, total simulation time, fault duration, fault models, etc.
3. Execution of the campaign. Each of the simulations defined in the previous phase are sequentially executed, detecting possible execution errors and generating simulation traces.
4. Results analysis. The simulation traces generated in phase three are compared to a “golden run” simulation trace (Gil et al. 2003), in order to study the behaviour of the model in presence of faults.

Like any other simulation-based fault injection tool, the main drawback of VFIT is the temporal cost of the injection campaign (phase three of Figure 1). This temporal cost is proportional to: i) the complexity of the model, ii) the complexity and size of the simulated workload, and iii) the number of fault injections to carry out. Distributed simulators can help reducing the time needed for one execution. This kind of simulators has better performance when simulating complex systems (Li et al. 2003), thus these simulators are not useful when dealing with multiple simulation executions of simple models. Another way to reduce the overall execution time is to execute each fault injection simulation in a different computer that acts as a simulation server.

The GSTF (Fault Tolerant Systems research Group) intends to develop a distributed version of VFIT according to the previous reasoning. Among the proposals that are

being evaluated is DVFIT-w, that makes use of web resources and protocols exclusively.

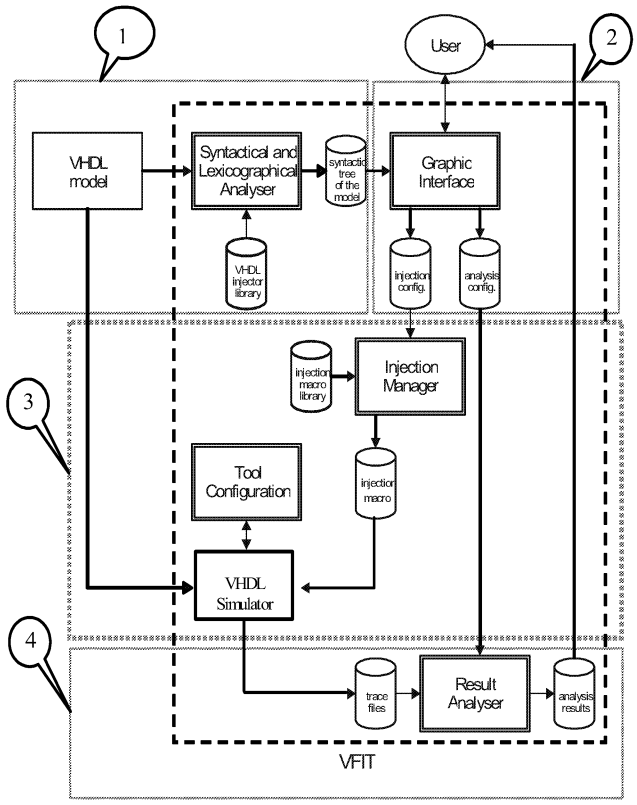


Figure 1. VFIT Main Components and Relationships, working phases are framed in boxes.

This paper presents firstly the different options the GSTF is taking into account for the development of a distributed version of VFIT, discussing their advantages and drawbacks. In second place, the paper focuses on the web-based version. A basic scheme of the tool is shown and, after that, the capabilities of its components are deeply described. Following the tool description, some utilisation results are discussed. They are compared with the local execution of VFIT sequential version. Finally, conclusions and future work are presented, including the study of different alternatives for the version presented in this paper.

DISTRIBUTED VFIT

VFIT is a tool that allows specifying and scheduling fault injection campaigns in an easy way. As stated before, its main drawback is the temporal cost of the fault injection campaigns. In order to reduce the response time, VFIT allows the user to divide the campaign into *sessions*, which only perform an interval of the total number of fault injections. Although this feature could be used for the workload distribution, it was not included in VFIT. Thus, the next natural step in the development of the tool is to implement this possibility for the automatic distributed execution of a fault injection campaign. VFIT versions that allow the distributed execution of a campaign will be named DVFIT (Distributed VHDL Fault Injection Tool).

On the other hand, the study of new techniques for distributed execution (GRID, web-based, agents, etc) would permit developing different DVFIT versions. Having these techniques in mind, our research group has discussed different possible implementations of the tool, which has led us to propose three different versions:

1. DVFIT-c: Custom client/server version. This is a traditional approach in the design of distributed applications. All application components are designed from scratch, including most of the communication protocols and data structures.
2. DVFIT-w. Distributed web-oriented VFIT version. The main idea is to use web-services as the communication layer among the components of the distributed VFIT. All the computation nodes execute a web server that offers services to register a VHDL model and to perform the fault injections.
3. DVFIT-g. Grid-oriented distributed VFIT. The idea is to use a grid system that executes the fault injection simulations. There is no need for explicit communication among the DVFIT components, apart from the campaign installation on the selected grid environment.

Table 1 presents a summary of the main advantages and drawbacks pointed out for each version. It has been decided to implement the three versions of DVFIT in order to check these assumptions. At last, a final version, based on results from performance tests, will be implemented.

The main goal of these three DVFIT versions is to reduce the execution time of fault injection campaigns, which corresponds to phase three of VFIT (see Figure 1). The distributed execution of phase four, and even of previous phases, is proposed for a second issue of the distributed tool. This paper presents the features of DVFIT-w, the first implemented and tested version, which focuses on the aim of reducing the execution time of fault injection campaigns.

DVFIT-W

DVFIT-w intends to distribute the work among several servers by using web servers and HTML and XML documents as main components. The final idea is to develop a system based on web services (SOAP) that can be the basis for several distributed simulation-based fault injection applications.

An additional objective of this first version was a hard deadline issue. In order to achieve this goal, we made two decisions that have greatly determined the structure of the application: to modify VFIT the least and not to use SOAP.

As it has been stated on the introduction, and it can be seen in Figure 1, an external simulator is used to perform the different simulations a fault injection campaign consists of. It is unfeasible to start the simulator from scratch for each simulation, since the time required to start the simulator is very high. To solve this problem, VFIT makes use of the

Table 1. Estimated Characteristics of the Three Versions of DVFIT.

	Advantages	Drawbacks
Custom Client/Server	Complete control over execution and communications Efficient use of bandwidth	Non-standard protocols Difficult to modify Entirely new versions for any desired change in its functionality Different versions for each operating system
Web Oriented	Use of well-known protocols and standards Use of middleware written in platform independent languages (tcl, java, etc) Independent web interface for the offered services Easiness to add new services Usable as a front-end for grid services	Protocols can be very little efficient in network utilisation
Grid Oriented	Distribution is transparent. Workload balancing is implemented on the grid system More calculation power Transparent and automatic adaptation to the characteristics of each node	Little control over execution system Trouble integrating new machines into server set Technology under development Different versions for each kind of operating system

scripting capability from ModelSim© to develop a small script that allows the communication between VFIT and the simulator without having to restart the simulator. This script is in charge of configuring the simulator to perform the execution, filtering the results and providing the main VFIT program with the error codes that can arise.

DVFIT-w takes advantage of this script to implement the client side of the web application. In this way, the main VFIT application is nearly the same, with minor changes, and communicates with this script, obtaining the same error codes and result files.

Figure 2 shows the block diagram of the client side of DVFIT-w. As it can be seen in the figure, it is almost the same as VFIT. The main difference is that simulations are not executed in the local machine. This prevents from having the commercial simulator in the local machine, allowing a better use of simulator licenses.

GET and POST messages from the HTTP protocol have been used to communicate the different components, by exchanging standard HTML documents. As previously stated, we will generalise these services by offering a SOAP interface in a later version.

Summing up, the structure of this first version is a three-layer architecture: client, middleware (the distributed execution script shown in Figure 2) and servers. The client application is the sequential VFIT and the middleware is a small application that makes use of the web servers as a client in order to achieve a distributed execution of the simulation. The following sections will describe in further detail the features of each one of these three components.

Server Side

The server side consists of a web server that implements the different required services. The decision of which web server to use is independent of the possible implementation of the client side, as long as the format and parameters of the messages that grant access to these services are observed. The offered services are:

- Model management: Installing, listing and removing models and retrieving model status.
- Simulate an installed model.
- Delivering and storing the results of simulations.
- Server state management: Dik space and Simulator version.

These services can be offered by two different kinds of interfaces:

- User-oriented: Web pages that ask the user, by means of forms, to introduce the required data to perform the desired operation. They might substitute some elements of the original VFIT or even the whole application.
- Application-oriented: Automatic data sending and receiving to and from the URLs that offer the services. There is no need for a user interface. This scheme may use a standard communication system with SOAP or may define a custom scheme that makes use of HTML documents for communicating.

This paper focuses on the application-oriented interface that will be used to automate the execution of fault injection campaigns. However, it can be easily seen how these services can be used to develop applications that will not need an executable file in the local machine, i.e. a true web application.

Sequential VFIT

As it has been previously stated and it can be seen in Figure 2, DVFIT-w keeps the original VFIT for phases 1, 2 and 4 (Figure 1). The middleware that substitutes the original simple script for communicating with ModelSim©

is in charge of executing the simulation in a distributed way.
The original code has been slightly modified to create and manage the new execution script. This modifications are needed to access some state variables that original VFIT uses to generate the experiments. Examples of such variables are the number of simulations to perform or the location of the model.

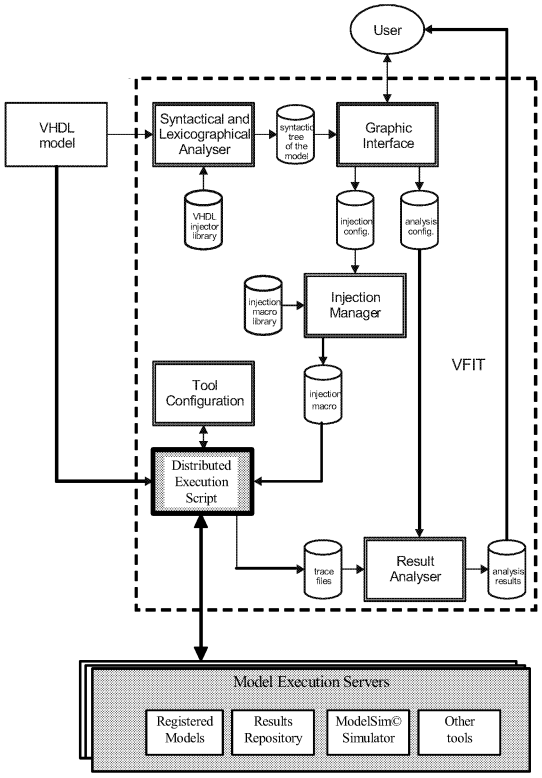


Figure 2. DVFIT-w Components and Relationships. (New components from original VFIT are shaded)

Middleware

The script that acts as middleware communicates as the original VFIT would expect the local execution script to do. However, it has a more complex task since it is in charge of:

- Registering the models in the servers.
- Synchronising the execution process.
- Detecting the free servers in order to perform a new simulation-based fault injection.
- Managing the errors that are not due to the model execution.
- Collecting data.

The possibility that the tool could run in an entire local way, i.e. with no simulation server at all, has been taken into account in the design of this script. In this case, the application has the original VFIT behaviour.
A possible improvement of this middleware would consist in integrating it into VFIT, that is to say, moving to a client/server architecture that can manage the possible errors in a better way and that can access information of the fault injection campaign the script is not able to access up to date.

DVFIT-W IMPLEMENTATION

All the new DVFIT-w components have been written in TCL. Both the Distributed Execution Script (Figure 2) and the web-services have been coded in this language. The main reason for the selection of this programming language, is the possibility of executing the scripts in the interpreter embedded in the commercial simulator (ModelSim©).
This implementation allows the server side application to keep the simulator running, to control any fault that can arise and to define the mode the simulator operates. Hence, not any process but ModelSim© must be running. This approach requires the fulfilment of the following conditions: 1) to obtain the required libraries for the different application components and 2) that these libraries perform their tasks in an efficient way.
The first condition has been fulfilled by using the following TCL libraries: TCL library 1.2 version, which is supplied with the commercial simulator; TclHttpd 3.4.2 version and MkZipLib 1.0 version.
Most of the problems that appeared during the implementation process were not due to the selected language but the commercial simulator and the way it creates the model files. The main problem has been the inclusion of absolute paths into the VHDL models files that ModelSim© uses, since it may cause a model not to simulate due to relocation problems.
The DVFIT-w execution follows the next sequence, assuming free error execution:

1. VFIT normal execution for model selection and campaign set-up (phases one and two of Figure 1).
2. VFIT executes the Middleware script and waits for a message.
3. The script reads the model parameters and the execution servers list. If this list is empty, the normal VFIT operation is performed (Gil et al. 2003).
4. The script creates a ZIP file with all the model files and directories. This file will be used in model registration.
5. The script registers the model in all the servers where the model was not previously installed.
6. The script sends a message to VFIT to get a simulation.
7. The script waits for a free server and sends the simulation to it.
8. The script asks VFIT whether another simulation is needed. If it is, then jump to step six. If not, go to step nine.
9. The script waits for pending simulations to complete.
10. The script downloads results from servers.
11. The script tells VFIT that the simulation execution has finished before exiting.

EXPERIMENTAL RESULTS

Tests have been based on a VHDL model of the well known PIC16C5x microcontroller running the Bubblesort algorithm on an integer array of 10 elements. Fault

injection instants and durations are randomly generated following a uniform distribution. Transient single bit-flips have been injected (Gil et al. 2003).
First, the results obtained with DVIFT-w were compared to VFIT results (DBench 2003)(Gracia et al. 2002). Both tools generated the same results and intermediate files. Tests also pretend to measure the speed-up obtained and to determine the impact of using TCL in the execution time of fault injection campaigns.
The different measured times are: *T.Sim* (Mean time to perform a fault injection simulation), *T.reg* (Time to register the zipped model in all the servers), *T.wait* (Time to wait for a free server when a new execution has been ordered) and *T.tot* (Time to carry out the whole fault injection campaign)
The campaigns have been executed with three different execution environments. The used computers have the following characteristics: Athlon XP1600+, equipped with 512 MB of RAM and running Windows XP©. The computers are interconnected via a 100 Mbits/sec Ethernet LAN.
The environments tested were a local VFIT sequential execution, and two different environments for DVFIT-w, with one server and with two servers. With one server, all the simulations are executed on the server machine.

Table 2. Measured Times for a Fault Injection Campaign with 2 Simulation Executions.

	Local	1 server	2 servers
T.Sim (ms)	6950	7125	7125
T.tot (ms)	21340	21000	12458
T.Reg (ms)	--	2500	2700
T.wait (ms)	--	15235	8123

Table 3. Measured Times for a Fault Injection Campaign with 10 Simulation Executions.

	Local	1 server	2 servers
T.Sim (ms)	6950	7125	7125
T.tot (ms)	70121	90123	40347
T.Reg (ms)	--	2550	2670
T.wait (ms)	--	85231	20807

The measured times are shown in Table 2 (a fault injection campaign with two simulation executions) and Table 3 (a fault injection campaign with 10 simulation executions).
The model registration is made as concurrent as possible using the event-oriented capabilities of TCL, so T.Reg is very similar with one and two servers.
As can be seen from results, the total execution time (T.tot) with one server and local execution are very similar, despite the time devoted to communications in DVFIT-w execution. This can be due to the fact that, while the server is performing a simulation (T.Sim), the client computer can prepare the next fault injection simulation. In VFIT, these two tasks are executed sequentially.
It is interesting to note that the total execution time (T.tot) has been considerably reduced when using two servers in both experiments. Using three computers we achieved a speed-up near to two, what is a great mark assuming that only two computers are executing most of the workload.

CONCLUSIONS AND FUTURE WORK

In this paper, we have presented DVFIT-w. This is the first distributed version of an earlier simulation-based fault injection tool called VFIT. This version is based in web technologies. Other two versions have been proposed, one based on custom client/server and another oriented to be executed in a Grid environment. These versions are under development.
The main objective of DVFIT-w is to reduce the execution time of simulation-based fault injection campaigns, by distributing individual simulation executions among several computers. These computers will execute a web-server with a known set of services accessible by standard protocols (HTTP, SOAP, etc.)
Validation tests show that DVFIT-w and VFIT obtain the same results. Moreover, DVFIT-w performance results are very promising. The achieved speed-up was near the maximum in the two servers tests. We hope that the speed-up will scale well with the number of servers.
Finally, we want to remark some improvements. First, the implementation of a SOAP based version would ease the development of new tools. A better implementation of services will reduce the response time.
The analysis phase of VFIT could be as time costly as simulations execution. New versions would include distributed versions of this phase for better overall performance. At present, distributed analysis tools, algorithms and protocols are under research.

ACKNOLEDGEMENTS

This work has been sponsored by the spanish project TIC-2002-02491 *Mejora de las técnicas de inyección de fallos en modelos VHDL*.

REFERENCES

DBench "Fault Representativeness", Deliverable ETIE2 of Dependability Benchmarking Project (DBench), IST-2000-25245, June 2002.

Gil, Daniel; Baraza, J.C.; Gracia, J. and Gil, P.J. 2003 "VHDL simulation -based fault injection techniques" Cap. 4.1 pp.159-176 in A. Benso, P. Prinetto, eds."Fault injection techniques and tools for VLSI reliability evaluation" ISBN 1-4020-7589-8 (2003)

Gracia, J.; Gil, D.; Lemus, L.G. and Gil, P.J. 2002. "Studying Hardware Fault Representativeness with VHDL Models", in Procs. XVII Conference on Design of Circuits and Integrated System (DCIS 2002), pp. 33-39, Santander, Spain, November 2002.

Li, Lijun; Huang, Hai and Tropper, Carl 2003. "DVS: An Object-Oriented Framework for Distributed Verilog Simulation". Proceedings of the Seventeenth Workshop on Parallel and Distributed Simulation (PADS'03)

Model Technology 2003,
"Modelsim SE User's Manual Version 5.8",

AGENT BASED AND NEURAL NET SIMULATION

A TOOL FOR SUPPORTING THE DEVELOPMENT OF CORRECT MOBILE APPLICATIONS BASED ON HIGHER ORDER π-CALCULUS

Andreea Barbu
University of Paris 12, LACL,
61 avenue du Général de Gaulle, 94010 Créteil, France
barbu@univ-paris12.fr

Keywords: mobility, mobile agents, formal verification, higher order π-calculus, Jini, simulation tool.

ABSTRACT

The purpose of this paper is to present the idea of our $HOPiTool$ simulation environment for mobile applications. We describe briefly the formal language the higher order π-calculus and the principals elements of the $HOPiTool$ tool that we use for the generation of mobile code. Through the simulation we can reason about the robustness and the fault detection of the system.

1. INTRODUCTION

Mobile agents are omnipresent in today's software applications and Java is a significant language to develop these agents. Such applications are hard to develop because of the concurrent and distributed aspect. Also code can migrate during the execution. Among other problems (security, dynamics of the links, connection and server failures) it is hard to test mobile applications in an environment like the Internet because we cannot be sure whether the problem is caused by the application itself or by the environment in which it runs.
The developers of mobile applications have to gain confidence in their solution during the construction phase. In order to do that, they need methods and tools that insure that desired properties are valid. In this context, the use of formal methods becomes a good procedure to provide a way to create a precise description of applications.
We use a formal specification language suited for mobile application, called higher order π-calculus [1] in order to obtain our simulation tool $HOPiTool$. Mobile applications written according to this formal language can be simulated with $HOPiTool$ who generates also java code. Considering a mobility support platform, in our case it will be the Java Intelligent Network Interface (Jini) [2], we can simulate the mobile application in a real network, using the code the tool generates.
Our achievements, so far, can be characterized by the mapping of an higher order π-calculus specification into a simulation model and the generation of code for a mobility support platform Jini. We found only one similar project that deals with object-based graph grammars [3]. There are some proposals of corresponding programming languages (e.g. KLAIM [4], Pict [5]), but on the level of specification and simulation of mobile applications, there is still no formal approach that would be largely used to simulate mobile systems. There are also projects in the literature that use formal verification techniques. Although planned, our tool does not address this point, so far. An interesting project that analyze concurrent systems over specifications written in the π-calculus is the tool Mobility Work-Bench [6].
Our major advantage of using a simulation model is the possibility of validating strategies for complex cases where the use of formal verification can lead to state explosion problem. With our $HOPiTool$ we can reason through simulation about mobile applications in presence of selected failures or tested properties.

2. THE FORMAL SPECIFICATION LANGUAGE HOπ

Through the use of a formal specification language we can create a complete description of a mobile system with a well-defined and complete syntax and semantics. The formal specification language used in this work is the higher order π-calculus, an extension of the π-calculus who was introduced the first time by R. Milner [7] and later by Sangiorgi and Walker [1]. The π-Calculus and its extensions are process algebra that focus on process mobility. Processes communicate using channels, which define the configuration of the system. Processes send a channel name in the monadic π-Calculus, tuples of channel names in the polyadic π-Calculus, and tuples of processes and channel names in the higher order π-Calculus ($HO\pi$).
Sangiorgi identifies a first-order paradigm and a higher-

order paradigm for mobility in process algebra. The notion of mobility in process algebra is realized by sending messages that change the communication interface between components of the system. The first-order paradigm allows ports or names to be transmitted as messages. After the transmission of a port, the communication can take place through this port. The higher-order paradigm allows processes (parameterized or not) to be passed as values in a communication. After a process has been transmitted, it can start its execution. This is a process-passing mechanism. Sangiorgi [1]proves that the expressiveness of higher-order and first-order π-Calculus are the same.
A higher-order π-Calculus process is given by the following syntax:

$P ::= \sum \alpha_i.P_i \mid P_1|P_2 \mid P_1 + P_2 \mid \nu\, x.P \mid [x = y]P \mid D\langle\widetilde{K}\rangle \mid X\langle\widetilde{K}\rangle$
$\alpha ::= x(\widetilde{U}) \mid \overline{x}\langle\widetilde{K}\rangle$

Where X is an agent (process) variable, $\langle\widetilde{K}\rangle$ stands for any tuple of agent or (channel) name, and $(\widetilde{U})$ stands for any tuple of variable or (channel) name. The constants D are defined as $D \stackrel{def}{=} (\widetilde{U})P$. Constants are to be seen as functions whose parameters can be processes or other functions and:

- $\overline{x}\langle K\rangle.P$ can send the name or process K via the name x and continue as P.
- $x(U).P$ can receive any name or variable U and continue as P with the received name substituted for U.We do not take into account sort of term in our algebra. This feature can be added later.
- in the composition $P_1|P_2$, the two components can proceed independently and interact via shared names or processes.
- $\nu\, x.P$ is called the restriction and means that the scope of name x is restricted to P.
- in the sum $P_1 + P_2$ either P_1 or P_2 can interact with other processes.
- The matching $[x = y]P$ denotes the activation of a process which is selected by other processes on depend of a condition ($[x = y]$) This operator $=$ is a boolean predicate defined in this algebra.

The difference between first-order and higher-order π-Calculus resides in the fact that parameters can be channels and/or processes in the higher-order π-Calculus, while in firs-order π-Calculus only channels can be passed as parameters.
Example: in $\overline{x}\langle P\rangle.Q|x(X).X$, once the interaction between the two processes has taken place, the resulting process is $Q|P$. Indeed, process $x(X).X$ was waiting for X to be sent along channel x, i.e., it was waiting for a process X defining its subsequent behavior.
The operational semantics is given in terms of a labeled transition system. There are three labels for the transitions: the silent step τ, the input action $x\widetilde{K}$ and the output action $\overline{x}\langle\widetilde{K}\rangle$.

Output action: $\overline{x}\langle\widetilde{K}\rangle.P \stackrel{\overline{x}\langle\widetilde{K}\rangle}{\longrightarrow} P$ means that after having sent message $\widetilde{K}$ (tuples of channels or processes) over channel x, process $\overline{x}\widetilde{K}.P$ behaves like P.

Input Action: $x(\widetilde{K}).P \stackrel{x(\widetilde{U})}{\longrightarrow} P\{\widetilde{U}/\widetilde{K}\}$ means that if message $\widetilde{U}$ (tuples of channels or processes) is sent over channel x, then the process $x(\widetilde{K}).P$, waiting for a process or channel name on x, receives it and replaces the process or channel name to $\widetilde{U}$, it then behaves like P, where all occurrences of $\widetilde{K}$ are replaced by $\widetilde{U}$. $\langle.\rangle$ stands for real parameters, while $(.)$ stands for formal parameters.
Interaction between two processes:

$$\frac{P \stackrel{(\nu\widetilde{K})\overline{x}\langle\widetilde{K}\rangle}{\longrightarrow} P', Q \stackrel{x(\widetilde{K})}{\longrightarrow} Q'}{P|Q \stackrel{\tau}{\longrightarrow} \nu\widetilde{K}(P'|Q')}$$

$\widetilde{K} \cap fn(Q) = \emptyset$ means that if an output action causes P to becomeP', and the corresponding input action causes Q to become Q', then P and Q in parallel become P' and Q' in parallel, and the private (bound) process $\widetilde{K}$ emitted by P becomes a private (bound) process of $P'|Q'$.
Equivalences: Bisimulation usually identifies processes with the same external behavior. Higher-order bisimulation identifies higher-order processes if their interactions with the environment are the same and if their internal processes are bisimilar.

3. THE SIMULATION ENVIRONMENT HOPITOOL

In order to simulate specifications written in $HO\pi$, we have implemented a simulation environment and described the mapping specifications to a simulation model. The simulation tool, is implemented in Java and consists of five parts. The rectangles in Figure 1. represent the functional entities, whereas the parallelograms are strategic data.

1. The graphic editor in order to be able to write a specification down.
   The objective is to represent by respecting the Unified Modelling Language (UML) notation, a specification where mobility is an essential characteristic. We propose five visualization windows for our editor:
   (a) an Extensible Markup Language (XML) visualization of our specification (Figure 2.), where

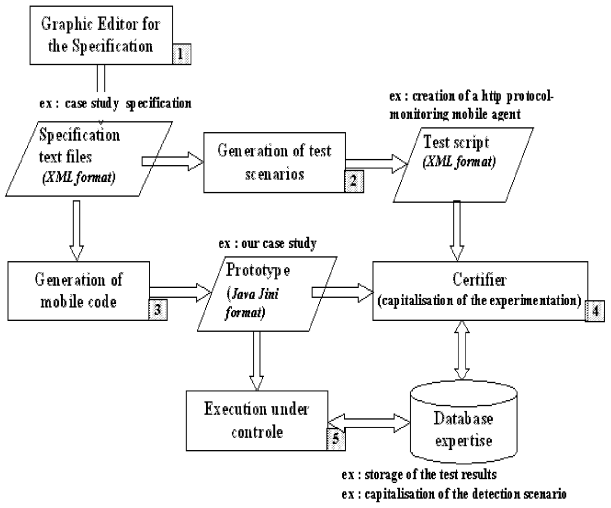


Figure 1. From Specification to Implementation

the XML file will be validate by the given HOπ Document Type Definition (DTD).

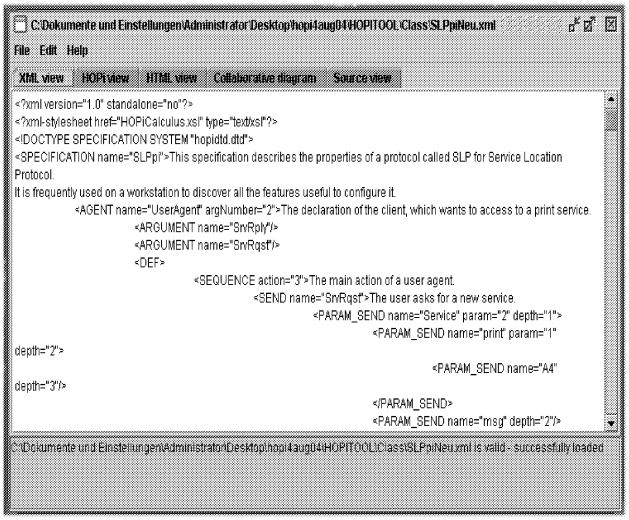


Figure 2. The XML View

(b) a π-calculus visualization respecting the style given by R. Milner.

(c) an HTML visualization (Figure 3.) in order to be able to show the documentation of each mobile agent

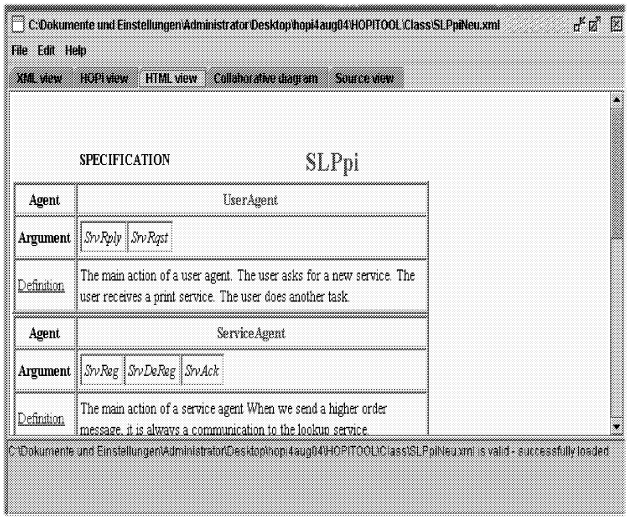


Figure 3. The HTML View

(d) an UML visualization using collaboration diagrams.

(e) a Java/Jini source code visualization

The XML file is the intermediate representation of a higher order π-calculus specification. It is used to create the charts and any change of a chart has consequences on this intermediate representation. It is an application of the Model View Controller (MVC) design pattern. The internationalization (we use the technics provided by Java) of the graphic interface aspects is an essential constraint. To developed global software, one of the great commandments of internationalization is to separate text, labels, messages, and other locale-sensitive objects from the core source code. This helps to maintain a single source code base for all language versions of our product. It also facilitates translations because all localizable resources are identified and isolated.
The major idea is to have a rigorous framework to control two important stages: the generation of test scenario and the generation of code.

2. The generation of test scenarios
From XML files describing a π-calculus specification, which means a set of mobile agents communicating through the services that they specified, the goal of this functionality is to generate test scenarios respecting the criterions given by the specification.
These criterions are meta rules useful for the generation of test sequences. The objective is to build tests which are used by the Certifier entity. Also, this test-set must answer to constraints of coherence, that

means to satisfy a common goal such as: "execute all the sequences". In other words our first goal is to build structural "white box" and not functional tests called more commonly "black box". Each output file describes a script of tests and also the essential measurement points for the work executed by the Certifier.

3. The generation of the mobile code
   This functionality is initiated by the graphic interface and it allows to obtain a code prototype from valid HOπ specifications. From XML files, describing a π-calculus specification, a set of source files is generated in order to obtain a prototype. The aim is to gain ability to do simulations and to validate the prototype through the tests which we defined before. More, HTML documents describing the documentation are generated by our tool, which takes in consideration the comments written within the specification. A specification consists of agents declarations. The whole of agents can be divided into two categories: mobile and non mobile. This constraint can be detected when an agent is send over a communication channel to an another agent. The mobile aspects take shape through the use of the technical framework Java Intelligent Network Interface (Jini) from Sun. The fundamental element is the utilization of Remote Method Invocation (RMI), which allows us to consider the network like object oriented. The services and clients in a Jini architecture can be integrated easier within the object interface. Jini disposes also of identifications and collect services, that is an important future when we want to develop a mobile agent system. The security questions will be treat through Java Cryptography Architecture (JCA) Norm from Sun.

4. The certifier
   This entity takes a Java prototype as entry but also a whole of test-scripts. Every test is applied over the prototype in order to determine if the prototype validates this test. We can distinguish two cases: the prototype satisfies the test and this simulation will enrich the data base or the prototype doesn't satisfy the test, an anomaly is lifted and a log file is saved. The log files are used in a "post-mortem" analysis phase, which is the basis of the Software Validation and Verification Report (SVVR). The analysis is not a part of this project.

5. Execution under control
   Every execution of a generated prototype is realized under control. Every anomaly compared with the acquired experience is done through a call-back alert from the sites where the conflict was detected.

Some important technical features are the Java 1.4.2 Swing Framework, the inputs and outputs are written in XML (with an external validation in form of a DTD file), Java Intelligent Network Interface Jini 2.0., Remote Method Invocation (RMI) and Oracle 9i in order to manage the data. Our $HOPiTool$ reads and generates Jini code from the XML file. The XML file is read and parsed using Java Architecture for XML Binding (JAXB) from Sun. A DTD document describes how these XML files should be structured and states that an XML agent file contains a series of agents which play together in a mobile system. Each agent has a name, a number of arguments and a definition of its form and its objective. We have defined each agent with the correspondents attributes and arguments. Based on this XML description of our mobile system and with help of our $HOPiTool$ we are able to generate Jini code for a further simulation of communicating agents in a network.

3.1. The Case Study

Our objective is to simulate a data acquisition system where a server sends an agent to two databases in order to collect data. The system is very simple, the communication occurs through data channels and listeners on server and database. Each database receives an instance of a mobile agent, and after receiving the data, an object with a safe result (it means that the collected data will be checked on safety and trustiness) will be send back to the server. The server can give the order to the agent to stop any action or to keep running on the database. Figure 4. shows the case study. At first we give a specification of our case

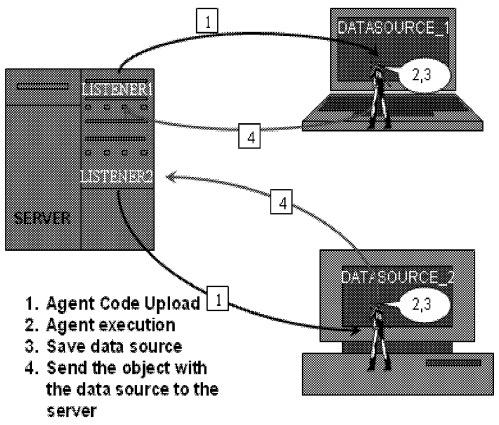


Figure 4. Data Acquisition System

study, using the HOπ-calculus in order to obtain a proper specification that can be used for our implementation.

$$
\begin{array}{l}
SYSTEM = \nu\,(listener_1, listener_2) \\
SERVER(listener_1, listener_2)\,| \\
DATASOURCE(listener_1)\,| \\
DATASOURCE(listener_2) \\
SERVER(listener_1, listener_2) \qquad = \\
\nu\,(channel_1, channel_2) \\
(\overline{listener_1} \;<\; AGENT(channel_1, listener_1) \;> \\
|LISTENER(listener_1))| \\
(\overline{listener_2} \;<\; AGENT(channel_2, listener_2) \;> \\
|\,LISTENER(listener_2)) \\
LISTENER(l) = l(SafeResult(r))\,.\,LISTENER(l) \\
AGENT(channel, l) = channel(d)\,.\,\bar{l}(SafeResult(d))\,. \\
AGENT(channel, l) \\
DATASOURCE(l) = \nu\,(a)\,l(AGENT(channel, l))\,. \\
(AGENT(channel, l)\,|\,!\overline{channel}(a))
\end{array}
$$

We specified as follows:

1. $SYSTEM$: describes our whole system with all the components.
2. $SERVER$: represents the server, which sends the $AGENT$ into the network to collect data information.
3. $LISTENER$: is related to the $SERVER$ and listen to request the data information sent by the $AGENT$.
4. $AGENT$: represents our mobile agent that are able to migrate to a $DATASOURCE$ in order to apply a job. The job is: find a database information and send this information as a safe one ($SafeResult$) to the $LISTENER$.
5. $DATASOURCE$ provides the data information needed by the $AGENT$.

In this specification the mobility is described by the exchange of agent over several channels. Using this knowledge, we are now able to define the XML file that we need for the $HOPiTool$. The generated Jini code consists of four java classes: the $SERVER$ class, the $DATASOURCE$ class, the $AGENT$ class and the $LISTENER$ class. These classes can communicate with each other in a network.

4. CONCLUSION

The tool is still in work, and we are working now to represent the UML and the Source Code view. The $HOPiTool$ generates automatically java code which can be use and simulates an mobile application on a mobile framework like Jini. We developed several mobile applications and the simulated behavior was coherent to the formal specification, indicating that the simulation code is complet. Also we are working on a test generator, which creates test sequences from initial π-calculus specification. Our intention is to obtain a test set, which will validate the generated code from $HOPiTool$. The test generator is configured with several criterions, which are essential to stop recursive unfolding in the agent definition.
The validation engine will be the final step of our project: it is an observer of the effects of the tests on the Jini prototype, generated by $HOPiTool$. Initially, our intention was therefore to develop an environment that allows the implementation of a mobile system of agents in Jini proceeding from a complet specification of the system given by the higher order π-calculus specification.

5. Author Bibliography

Andreea Barbu is a PhD Student in Computer Science. The PhD programm is a cooperation programm between the University Paris12 in France and the Carl v. Ossietzky University in Oldenburg, Germany. She graduated from the Carl v. Ossietzky University in Oldenburg, Germany in 2000 with a diploma in Computer Science. Current research topics are related to mobile computing, concurrent systems and formal languages.

6. REFERENCES

[1] D. Sangiorgi, "From π-Calculus to Higher-Order π-Calculus – and back", in Proc. TAPSOFT '93, LNCS 668, Springer Verlag.

[2] N. N. "Jini Network Technology", Sun Microsystems, Inc., California, USA, 2001.

[3] D. E. Perry, and G. E. Kaiser, "Simulation of Mobile Applications", IEEE Trans. Software Eng., vol. 17, no. 3, pp. 283-295, Mar. 1991.

[4] R. De Nicola, G. Ferrari, R. Pugliese, "Klaim: a Kernel Language for Agents Interaction and Mobility" IEEE Transactions on Software Engineering, 24(5):315-330, 1998.

[5] B. C. Pierce, and D. N. Turner, "Pict: A Programming Language Based on the Pi-Calculus", Technical Report TR476: http://www.cs.indiana.edu/cgi-bin/techreports/TRNNN.cgi?trnum=TR476

[6] B. Victor and F. Moller, "The Mobility Workbench for the pi-calculus", In D. Dill, editor, Proceedings of CAV'94, vol 818 of Lecture Notes in Computer Science, Springer Verlag, 1994.

[7] R. Milner, "Communicating and Mobile Systems: the π-Calculus". Cambridge University Press.

SIMULATION OF BUSINESS PROCESS IMPROVEMENT BY A CUSTOMISED GENETIC ALGORITHM

Cornelia Richter-von Hagen, Dietmar Ratz, and Wolffried Stucky
Institute for Applied Informatics and Formal Description Methods (AIFB)
University of Karlsruhe
D-76128 Karlsruhe
Germany
Email: cri,dra,wst@aifb.uni-karlsruhe.de

KEYWORDS

Business Applications, Process-oriented Optimisation, Operations Research, Genetic Algorithms.

ABSTRACT

In this paper we study the capabilities of a Genetic Algorithm (GA) for Business Process Improvement (BPI) with one important aspect being the prevention of critical conditions of the process which may be imposed by radically changing surroundings. The often appearing presence of multiple evaluation criteria requires an analysis of Multi-Objective Genetic Algorithms (MOGAs) to find acceptable Pareto-optimal solutions for the problem as 'Trade-offs'. For our case, a particular GA — the Multi-Sexual GA (MSGA) — will be applied to a special class of business processes.

1 BUSINESS PROCESS IMPROVEMENT

Business processes can be found in any organisation. Due to a fast changing economic world the surrounding conditions for business processes change and therefore necessitate flexible and adaptive processes.

Definition 1 Business Process
A business process is a sequence of activities aiming at the creation of one or more products or services with a value for a customer. It is started and finished by one or more events. As it proceeds in an organisation there is an underlying organisational structure. Richter-von Hagen and Stucky, 2004
Furthermore, the activities usually need one or more resources (like people, processors, data, software, etc.) that belong to predefined resource classes.

Business processes can be structured, semistructured and unstructured. Structured processes are completely predefined. There are fixed and non changeable rules for the execution of every activity. A structured process is repeatable as often as needed. Semistructured processes contain structured parts and non structured parts. Unstructured processes are completely unpredictable.

There are two main methods to improve business processes. One is known as Business Process Reengineering (BPR) and is the revolutionary method. The other one is known as Business Process Improvement (BPI) or also as Continuous Process Improvement (CPI) and is the evolutionary method. Whereas BPR is a fundamental rethinking and a radical reorganisation of the processes, BPI aims at a continuous and incremental improvement of business processes.

Definition 2 Business Process Improvement
Business Process Improvement is an evolutionary method to improve business processes aiming at a continuous and incremental improvement of processes.

The use of performance indexes enables the evaluation of the improvement of a process. Performance indexes are described for example in Gadatsch, 2001. There they are classified (concerning business processes) into process oriented and resource oriented and also into time oriented, value oriented and quantity oriented indexes as shown in Table 1.

Process oriented	Resource oriented
Time oriented	Time oriented
throughput time, execution time, service time, idle time, ...	operation time, idle time, inactive time, ...
Value oriented	Value oriented
costs of process, ...	used capacity costs, idle time costs, ...
Quantity oriented	Quantity oriented
executed activities, (not) exec. activities, ...	object input, output, object stock, ...

Table 1: Performance Indexes for an Evaluation

Further classes of performance indexes besides process oriented and resource oriented can be product and/or service oriented. Furthermore, besides quantity oriented indexes could be considered quality oriented performance indexes.

The surroundings of a business process can either depend on the business area in which the process is implemented or they can be domain independent. Examples are preconditions for the process like dependencies between activities or resources, resource restrictions or strategic decisions of a company, like a changed quality standard or an increasing role of innovation.

The change of surrounding conditions can influence one or more performance indexes and therefore lead to critical and undesirable values. Critical conditions can be all important oversteppings of fixed values for certain performance indexes. Consequently it will be necessary to define all important performance indexes with their critical values in advance.

2 TRADE-OFF AND PARETO SOLUTIONS

Business Process Improvement generally requires the consideration of more than one performance index. Therefore BPI is one of the many real-world problems that depend on a handling of multiple criteria. These can be described by a set of objective functions with additional constraints. Usually it will not be possible to improve all of the objective function values without worsening some others. This phenomenon is referred to as Trade-off and is demonstrated in Figure 1. The improvement of one criterion can result in a deterioration of one or more others. The aim is to best increase the imagined area between all objectives.

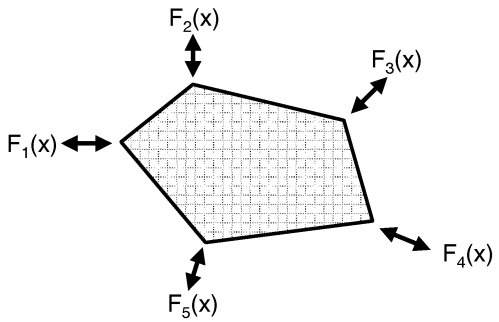


Figure 1: Trade-off of Performance Indexes

Multiple criteria are treated in multi-objective optimisation.

Definition 3 MULTI-OBJECTIVE MINIMISATION PROBLEM (MOMP)
Minimise $F(x)$,
with $x \in \mathrm{R}^n$,
and $F(x) = (F_1(x), F_2(x), ..., F_k(x))$
consisting of k ($k \geq 2$) objective functions.

There does not necessarily exist one unique optimal solution. Instead there is a set of solutions, called Pareto(-optimal) or nondominated solutions, better than all other solutions of the searchspace.

Definition 4 PARETO-OPTIMAL (OR NONDOMINATED) SOLUTION
A vector $x^* \in \mathrm{R}^n$ is called Pareto-optimal or nondominated solution of a MOMP, if
$\nexists\, x \in \mathrm{R}^n$
with $F_i(x) \leq F_i(x^*) \quad \forall i = 1, ..., k$ and $x \neq x^*$.
Vector x^* is nondominated by any vector x.

The other solutions are referred to as dominated solutions. Pareto-optimal solutions are generally not optimal for any of the multiple criteria.

3 GA METHODOLOGY

Genetic Algorithms are heuristical methods that can be used to solve a very broad class of optimisation problems under consideration of the problem's environment.

The solutions of the problem are encoded in a certain representation and can be evaluated with respect to the environment, i.e. an evaluation function is needed that represents the environment's evaluation of a proposed solution. Therefore GAs require three main components to be defined: a representation, an evaluation, and some operators to be applied, basically the selection, crossover, and mutation operator.

The universal GA can be described as follows:

1. generate a population of potential solutions, called individuals;
2. evaluate each individual out of the population; assign fitness values;
3. selection of mating members from the population dependent on their fitness value;
4. crossover and mutation of the mating members;
5. replace the entire population;
6. if stop condition satisfied then stop, else goto step 2.

Some GAs for multi-objective optimisation have been presented already. The first approach referred to as Vector Evaluated GA (VEGA) was published by Schaffer Schaffer, 1985, and later approaches such as the Niched Pareto GA (NPGA) by Horn et al. Horn et al., 1994, and the Nondominated Sorting GA (NSGA) by Srinivas and Deb Srinivas and Deb, 1994.

There are two further interesting approaches. One is the Strength Pareto Evolutionary Algorithm (SPEA), presented by Zitzler and Thiele is delivered with a comparison to the algorithms mentioned above Zitzler and Thiele, 1999. This comparison gives the following ranking considering some universal problems: SPEA, NSGA, VEGA, NPGA.

Another one — the Multi-Sexual GA (MSGA) — presented by Lis and Eiben Lis and Eiben, 1996 was already successfully applied in Bonissone and Subbu, 2003 to a flexible manufacturing problem.

4 GA APPROACH TO BUSINESS PROCESS IMPROVEMENT

The complexity of real BPI problems results in an extremely complex and time-consuming optimisation task, characterised by high-dimensional, nonlinear dependencies between process variables $x_1, ..., x_n$ and the objective function values $F = (F_1(x), ..., F_k(x))$, hence GAs can be applied to solve such problems Bäck, 2002.

Some examples considering the special class of structured business processes with constrained resources (there exists at least one resource class containing less resources than required by the activities of the process at the same time) were implemented in Delphi5 and tested.

First, a particular GA is applied to the special class of structured business processes with constrained resources considering one single criterion. Thereafter, for the multi-objective BPI problem the Multi-Sexual Genetic Algorithm is applied, because of good experiences (see e.g. Eiben et al., 1994 and Bonissone and Subbu, 2003). One important advantage of multi-parent crossover operators is that they are more explorative and less sensitive to premature convergence.

The main characteristics of the Multi-Sexual Genetic Algorithm (MSGA) are:

- as many sexes as optimisation criteria are used and each individual is evaluated according to the optimisation criterion related to its sex;
- the multi-parent crossover (requiring one parent from each sex) is applied to generate offspring;
- a set of nondominated solutions is updated during execution; this set is the output.

The design choices made for the special class of structured business processes with constrained resources are described below:

- The representation of solutions consists in ternary strings using the alphabet $\{-1, 0, 1\}$ with a sex marker (here we used an integer) at the last string position.
- Three criteria (e.g. execution time, cost and resource usage) as evaluation functions were considered.
- The operators *Selection*, *Crossover*, and *Mutation* were adapted according to Lis and Eiben, 1996 and applied to the ternary representation.

Changed surrounding conditions of the business process can result in critical conditions. The prevention of these demands another operator not belonging to the main GA operators that checks the fixed critical values of the performance indexes.

5 CONCLUSION AND FUTURE WORK

The customised MSGA was successfully applied to one of the manyfold BPI problems. It remains to discover the potential of the described MSGA as well as other MOGAs featuring specially customised operators to other BPI problems. Open questions are the fitting of MSGAs to semi- or even unstructured processes. Because of the occurrence of many constraints (maybe for the critical conditions) it should be worthwhile to investigate the implementation of a rule base. For diffuse constraints even a fuzzy rule base maybe a promising approach.

Furthermore, it is interesting to extend our version of the MSGA to the control and the automated improvement of business processes. This can be achieved by first using data analysis and data mining techniques on collected data. Afterwards GA techniques can improve the business process which again delivers data for another analysis and so on. This process can be called self-adaptation and promises an automated adaptation without user interaction.

REFERENCES

Bäck, T. (2002). Adaptive business intelligence based on evolution strategies: some application examples of self-adaptive software. *Information Sciences*, 148:113–121.

Bäck, T. and Willmes, L. (2003). Evolutionäre Algorithmen für die Optimierung dynamischer Produktionsprozesse. In Abel, D., Harbach, F., and Jumar, U., editors, *Proc. GMA-Kongress 2003, Automation und Information in Wirtschaft und Gesellschaft*, volume 1756 of *VDI-Berichte*, pages 995–1002, Düsseldorf. VDI-Verlag.

Bagchi, T. P. (1999). *Multiobjective Scheduling by Genetic Algorithms.* Kluwer Academic Publishers, Boston, Dordrecht, London.

Bonissone, S. and Subbu, R. (2003). Exploring the Pareto Frontier using Multi-Sexual Evolutionary Algorithms: An Application to a Flexible Manufacturing Problem. Technical Report 2003GRC083, GE Global Research.

Deb, K. (2001). *Multi-objective optimization using evolutionary algorithms.* Wiley, Chichester, Weinheim.

Eiben, A. E., Raué, P.-E., and Ruttkay, Z. (1994). Genetic algorithms with multi-parent recombination. In *Proceedings of the 3rd Conference on Parallel Problem Solving from Nature*, pages 78–87. Springer.

Freitas, A. A. (2002). *Data mining and knowledge discovery with evolutionary algorithms.* Springer, Berlin, Heidelberg.

Gadatsch, A. (2001). *Management von Geschäftsprozessen.* Vieweg, Braunschweig/Wiesbaden.

Horn, J., Nafpliotis, N., and Goldberg, D. E. (1994). A niched pareto genetic algorithm for multiobjective optimization. In *International Conference on Evolutionary Computation*, pages 82–87, Orlando, FL.

Lis, J. and Eiben, A. E. (1996). A Multi-Sexual Genetic Algorithm for Multiobjective Optimization. In Fukuda, T. and Furuhashi, T., editors, *Proceedings of the 1996 International Conference on Evolutionary Computation*, pages 59–64, Nagoya, Japan. IEEE.

Richter-von Hagen, C. and Stucky, W. (2004). *Business-Process- und Workflow-Management. Prozessverbesserung durch Prozess-Management.* Teubner, Stuttgart, Leipzig, Wiesbaden.

Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In *Proc. of the 1st International Conference on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum Associates, Inc.

Srinivas, N. and Deb, K. (1994). Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248.

Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271.

AUTHORS' BIOGRAPHIES

CORNELIA RICHTER-VON HAGEN was born in Cottbus, Germany and went to the Technical University of Chemnitz, where she studied mathematics, business sciences and philosophy, and to cole Nationale des Mines de Nancy, France, where she studied industrial engineering. She obtained her degree in mathematics from the Technical University of Chemnitz in 1999. Since the end of 1999 she has been working as scientific assistant and lecturer at the Institute of Applied Informatics and Formal Description Methods (AIFB) of the University of Karlsruhe (TH) as well as at the University of Cooperative Education in Karlsruhe in the field of applied informatics and business process and workflow management. Her fields of research also comprise decision support systems and the application of fuzzy sets.

DIETMAR RATZ was born in Karlsruhe, Germany and went to the University of Karlsruhe (TH), where he studied mathematics. He obtained his doctoral degree of mathematics in 1992. He then worked as a postdoctoral scientific assistant at the Institute of Applied Mathematics until he obtained his postdoctoral lecture qualification in 1998. Since then he has been Associate Professor for Applied Informatics at the Institute of Applied Informatics and Formal Description Methods (AIFB) at the University of Karlsruhe (TH). He is lecturer in the fields of (object-oriented) programming languages and numerical mathematics at the University of Karlsruhe (TH) as well as at the University of Cooperative Education in Karlsruhe. His fields of research also comprise verified computing, automatic differentiation and global optimisation.

WOLFFRIED STUCKY was born in Bad Kreuznach, Germany and went to the Saarland University at Saarbrücken, where he studied mathematics. He obtained his doctoral degree of mathematics in 1970 and worked then for the pharmaceutical industry in the areas of biometry, statistics and data processing. Since 1976 he has been full professor for Applied Informatics and his current position is head of the Institute of Applied Informatics and Formal Description Methods at the Department of Economics and Business Engineering at the University of Karlsruhe, Germany. The fields of research comprise data base and information systems, workflow management systems, strategic IT-planning and organisation, eLearning, tele-teaching and cooperative learning. He was serving as CEPIS (Council of European Professional Informatics Societies) president until the end of 2003.

SIMULATION ROUTINES FOR TEACHING AND TRAINING THE POWER SYSTEM MANAGERS

Eugeniusz M. Sroczan
Institute of Electric Power Engineering,
Poznañ University of Technology
ul. Piotrowo 3A
60-965 Poznañ, Poland
E-mail: sroczan@put.poznan.pl;

Andrzej Urbaniak
Institute of Computing Science
Poznañ University of Technology
ul. Piotrowo 3A
60-965 Poznañ, Poland
E-mail: urbaniak@put.poznan.pl

KEYWORDS

Electrical engineering, ecology, man-in-the loop simulation, simulation in education, AI decision support.

ABSTRACT

Energy complex effects the environment due to technical functions fulfilled and interactions among their participants – providers and consumers of energy resources. The simulation of some decisions of power system managers was possible with the use of a formal descriptions of different states that occurred in the relationship among energy providers and consumers as well as a management policy and energy consumption gained from renewable sources of energy. Some routines, used by power system managers, take into account the relationship among expected costs of consumed primary energy, environment and the income specified for environment's users. Providing that circumstances and environment protection policy are given. The results of the simulation describe some solutions, in a quality and quantity manner, which are proposed by the participants to transactions made on the energy stock in the form of contracts. Routines built in the simulator for costs calculation based on multi-criterion decision support system and artificial intelligence modules, are used to teach and training the energy manager to meet the requirements of energy market. Some routines of teaching are described in this paper.

INTRODUCTION

It is obvious, that electric power plants (PP) effect the natural environment depending on the technology of primary energy conversion to electric one. Thermal power plants, fed with coal, generate electric power and simultaneously combustion products. Additionally they raise the temperature of water in rivers or lakes due to pumped out cooling water. If the primary energy is gained from gas or oil, the water as combustion residue is emitted (from plant) into atmosphere too. The same problem concerns the conventional heating central plants (HC) installed in urbanized areas and thermal (also nuclear) power plants with cooling towers. Nowadays, in some countries, the distributed system of energy converting is applied as a more economic one than a central method of electric power supply. Additionally it generates the heat (co-generation) for central heating purposes. That method is specific for the local energy market (Malko 1997). It means that primary energy supplying a local power plant is gained from local sources. These sources are as follow: hydro power, energy of wind (wind turbine-WT), geothermal heat, thermal co-generation, bio-gas, biomass and photo-voltaic cells. Some relationship among energy sources, energy providers and energy consumers are shown in the figure 1 (Sroczan and Urbaniak 2003). The local energy market, as a set of energy sources and purchasers of local significance, is embedded in the national power system (PS). It allows to avoid the result of power failure occurring in the local power (or electric-heating) plant. Therefore it must contribute with other participants to the PS, with respect to economic and technical constraints. The crucial function is realized by the energy manager of the PS, which is tuning the problem of commitment, optimal dispatching and managing the power flow due to a number of decision concerning the topology of network and a number of unknown factors caused by customers of energy. The problem of balancing the power demand and power generation (Sroczan 1995), in *t-th* time period (moment), takes into account following constraints:

$$\sum_i P_{git} = P_{dt} + \sum_i \Delta P_{TDNit} \tag{1}$$

$$P_{gi\min} \le P_{git} \le P_{gi\max} \tag{2}$$

$$V_{Tj} \le V_{Dj} \tag{3}$$

where: P_g - generated power, P_d - power demand of PS; ΔP_{TDN} - power losses in transmission and distribution network, V_D,V_T – disposed and processed volume of water in j-th HP plant, i,*j*- power plant number, *t*– discussed time period.

The PS manager arranges the set of generating sources with respect the characteristic of generated cost of each unit of PP (or long term contracts) in accordance with the power load versus the duration time of load. Renewable energy sources, usually allocated in local energy market decrease the volume of demand power and energy from national PS. Production volume of some power plants, especially HP plants, is limited due to disposed volume of water.

The multi–criterion evaluation of the hydro–energy complex has an impact the state of environment. It shows that the power generating is not a main benefit from this complex.

The change of point in the economical estimation of the investment in building and HP (or WT plants) exploitation

can result in the increase of the economic attractiveness of investment, depreciation or even regret of it all. The real value of energy is balanced on the energy stock in accordance with the contracts between producers, providers and consumers. To solve the problem of cost minimization of generated energy neural networks are applied.

ENERGY MARKET INFLUENCING THE NATURAL ENVIRONMENT

The energy stock and local energy markets effect natural environment due to converting primary energy from natural source, transmitting of power and energy and converting by end users the electric (heat) energy to desired medium. It means that the energy consumers as well as energy producers and providers effect the natural environment. However, the contracts for energy exchange and especially the decisions of power system managers effect significantly the environment in the given circumstances.

All contracts from system market and especially on balancing market are verified (checked) by manager's office system from technical and economic point of view. The main objective of the office is to judge the technical accuracy of stock contracts among power plants and wholesale energy providers and also preserve the solution given by

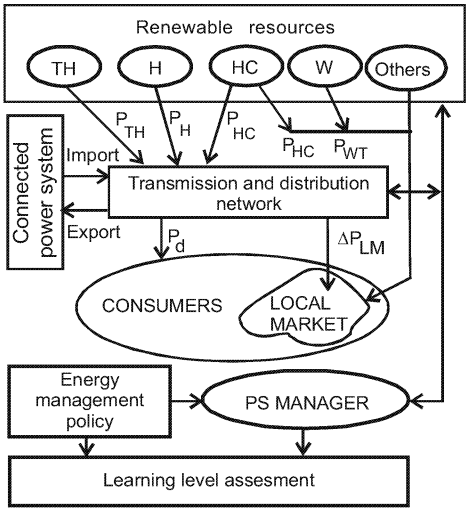


Fig 1. Basic relationship among sources and consumers of energy; TH - thermal PP (conventional and nuclear), H -hydro PP, W -wind turbines, HC-heat central, Others- photo-voltaic-cell, geothermal HC, bio-gas, biomass.

equation (1) in discussed time period.

The main tasks that should be considered by power system manager are following:

- Realize the short-term contract for energy transmission from the other system (countries) even if the price of the energy covers only the variable part of generated costs.
- Electric power and energy dispatch, as a result of energy exchange contracts, should preserve stability and safety of the power system as well as the quality of delivered energy.
- Allocation of the power and the energy resources to meet the contract's requirements.
- Balancing the power (energy) delivery and demand in the power system in each moment of considered time period in accordance with balancing and spot market.
- Current operation of connection lines to neighbor power system

The structure of simulation of the costs of power generation is shown in the figure 2. The proposed model of managing the PS is based on the maximization of expected delivery income (EDI) calculated for the k-th energy provider, which is described as Ω - a set of committing power plants and final distributors (Sroczan 2000, Sroczan and Urbaniak 2003).

Expected income, calculated for this set take into account following relationships: power plant, energy and power exchange, provider, distributor, consumer, in j-*th* hour of considered time period of calculation T:

$$\max\{EDI_k\} = \max\left\{ \sum_{i,k \ni \Omega} \sum_{t \ni T;} \left[\left(a_{kt} + b_{kt} P_{ikt} + c_k P_{ikt}^2\right) + \left(-\left(a_{it} + b_{it} P_{ikt} + c_i P_{ikt}^2\right)\right) \right] \right\} \quad (4)$$

$$\sum_i P_{git} = \sum_k P_{kt} + \sum_i \Delta P_{FLit} \quad (5)$$

where: EDI_k - income of k-*th* energy provider which is contracted power and energy from i-*th* power source (PP), belonging to set Ω; a_i, b_i, c_i, a_k, b_k c_k - characteristic's factors describing the relationships between hourly exploitation costs of i-*th* power plant and k-*th* distributor, defined for set Ω; P_g – generated power; ΔP_{FL} – power losses in given network branch of PS.

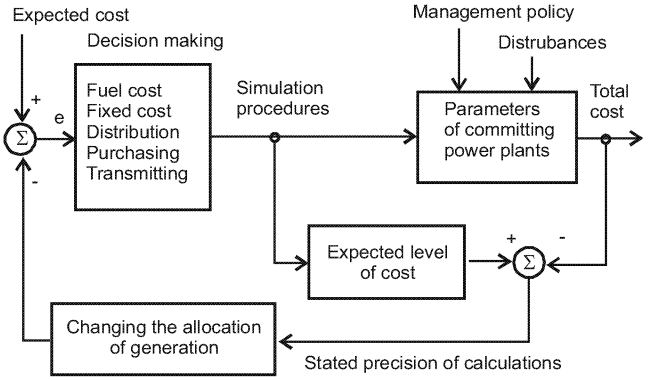


Fig. 2. The algorithm of cost simulation.

Equation (5), as modified (1), describes constraints of power balance in PS including generated power and power losses in transmission and distribution networks with regard to power transit.

SIMULATION OF POWER AND ENERGY COSTS

The exercise price of delivered energy is calculated as average value, defined in the real-time mode by PS operator office. That value is obtained (Sroczan 2000) in the *t* - moment of considered time period when power balance meets the constraint (2) and (3):

$$P_{gt} = \sum_i P_{THit} + \sum_j P_{Hjt} + \sum_k P_{HCkt} + \sum_l P_{LMlt} + \sum_p P_{Wpt} - \sum_m \Delta P_{TLmt} - \sum_n \Delta P_{DLnt} - P_{EXPt} + P_{IMPt} \quad (6)$$

where: P_{THi} - power generated by thermal PP, P_{Hj} - power generated by hydro PP, P_W- power generated by WT, P_{HCj}- power co-generated by heating central (local market embeded in PS), P_{LMl} - power from distributed renewable sources (local market), $\Sigma\Delta P_{TLm}$ - transmission power losses, $\Sigma\Delta P_{Dn}$ - distribution power losses, P_{EXP} - export of power to others PS, P_{IMP} - imported power from other PS, *i,j,k,l,m,n,p* - numerous of sources and lines.

The essential level of the price of power C_E (and energy) is effected by PS manager decision, if the decision is optimal the costs will change each time *t* according to the following relationship:

$$\min C_{Et} = \left\{ \begin{array}{l} \sum\limits_{i} \left[\begin{array}{l} c_{cfti}\left(P_{THcti}\right) + c_{cpti}\left(P_{THcti}\right) + \\ + c_{ai}\left(P_{THcti}\right) + c_{cwi}\left(P_{THCti}\right) \end{array} \right] + \\ + \sum\limits_{j} \left[c_{nftj}\left(P_{THntj}\right) + c_{cwj}\left(P_{THntj}\right) + c_{nwj}\left(P_{THntj}\right) \right] + \\ + \sum\limits_{k} c_{TDk}\left(P_{gtk} + P_{EXPtk} - P_{IMPtk}\right) \end{array} \right\} \tag{7}$$

where the unity costs are defined as follow: c_{cf} – conventional fuel, c_{nf} – nuclear fuel, c_{cp} – emission of combustion products , c_a – solid wastes, c_{cw} – cooling water, c_{nw} – nuclear wastes, c_{TD} – transmission and distribution, *i,j,k* – index number.

It is assumed that the cost of power and energy generated in PS can be calculated for the thermal and hydropower plants assuming the equivalent cost's factor of energy generated by hydro-power and wind plants. The cost of the power and energy delivered upon the bilateral contracts effects the local market, but if demanded power is greater then power capability of this market the deficit power is imported from PS. The load of PS is increased by ΔP_{LM} and therefore effects the cost of PS and effects the natural environment. The transmission and distribution networks topology has to be taken into consideration using the penalty factors (Sroczan 1995, Das and Heimes 1979, Stoner and Wankel 1986, Chowdhury and Saifur Rahman 1990).

SIMULATION OF PS MANAGER DECISIONS

In case of PS manager we can analyze this process as a two level hierarchical approach. First level regards the power dispatch between thermal and hydro power plants respect to all requirements and restrictions. Taking into account many connections with connected power system all external restrictions must be fulfilled completely. Thus, main goal of the system management is to obtain the minimum overall maintenance cost in respect of the environmental restriction namely listed in the figure 2 and factors respected in eq. (2) and (5). The mathematical model of this problem may be formulated as a ε - constrained problem. A method of solving that kind of problems was proposed by Das and Hames in 1979. The problem is more complicated when more realistic situation is assumed in which the chosen parameters will be imprecise and unknown or dynamic changed (Abbasy and Al-Hasawi 2003, Yoshihara S. at al 2003).

The second level of the decision making process concerns the balancing energy market. The manager of the power system decides about the power value, which must be generated or shut down in the system regarding to the different kinds of the committing power plants. The operation cost of the different power plants in this moment are not the same as the first decision level. The manager is searching and queuing the cost

$$\min_{j}\left\{ EMV\left(C_j\right) = \sum_{i=1}^{n} EDI_j\left(S_i\right) \cdot p\left(S_i\right) \right\} \tag{8}$$

where: $EMV(C_j)$- expected monetary value of decisions in a given probability of power demand depending on the cost of the j-*th* considered power source (PP).

In practice, the PS manager, as decision maker, ought to be equipped with the computer decision support system processing special pre-calculated pay-off table for all possible scenarios . The number of possible scenarios is not so great with respect to power treated as a discrete value. The strict power balance of the PS is realized by operating the system for the frequency control for active power balance and voltage control for reactive power.

The aspects of validation should take into account both terms of analysis: the technical and economic effects. In some countries the energy exchange is defined as electric power stock. In this case the thermal power and energy, destined for heating purposes are considered as a product dedicated to the local energy markets. The proposed criterion function (8) supports the decisions maker for optimization of the power plant load in considered PS to meet the constraints of contracts with maximal efficiency of energy converting.

The structure of simulation of the cost, shown in the figure 3., is obtained for the set of thermal power plant, equivalent factors for HP and WP sets and is defined as:

$$\kappa_{it} \frac{\Delta C_{WP}}{\Delta P_{WP}} = \gamma_t \frac{\Delta W_{HPit}}{\Delta P_{it}} = \frac{\Delta C_s}{\Delta P_s} = \lambda_t \tag{9}$$

where: ΔW - increment of consumed water, ΔP – increment of generated power, and κ represents an increment of cost of the power generated in wind turbine plants, γ - a system prise of water that is processed in each HP.

Partial optimisation of hourly load program of hydro power plant, in the given period of time *T* and expecting the disposal and consumed volume of water, is calculated upon the base of γ_t and κ_t.

Boundaries and restriction factors, respecting the environment effect are defined as penalty factors:

$$p_{cikj}\left(P_{gikj}\right) = \frac{1}{1 - \frac{\Delta F_{ikj}\left(P_{gikj}\right)}{\sum\limits_{i}\left[\Delta F_{kj}\left(P_{PSj}\right)\right]}} \tag{10}$$

where: p_{ck} – penalty coefficient describing *k-th* parameter, $\Delta F_{ik}(P_{gi})$ – increment of costs due to respected k-*th* restriction, i – number of the respected PP, *j* –a number of the covered range of load power.

In the real circumstances of the power system load, the increment of costs generating (for n-*th* subset of committing units) is optimal when:

$$\frac{\partial C_{THt}}{\partial P_{nt}} = \lambda_t = \gamma_t \frac{\partial W_{HPt}}{\partial P_{mt}} = \kappa_t \frac{\partial W_{WPt}}{\partial P_{WPt}} \tag{11}$$

Equation (11) describes the optimal state of generation and delivery in the PS. The increase of water consumption in the hydro power plant is proportional to variation of costs

generating of considered thermal power plants set.

ASSESMENT OF EFFECTIVENESS OF TRAINING THE OPERATOR

The applied model of knowledge base of the manager is essential for simulation the affects of decision made by the system operator. That attempt to support the decision of PS manager is acceptable by the system operator thanks to user friendly description of problem.

Using the described process of optimisation it is possible to obtain the minimal costs of energy and optimal allocation of the given energy resources.The process of training the NN is described as iterative tuning of coefficient w_i of each inputs x_i to meet the value $y(k)$ at the output of NN.

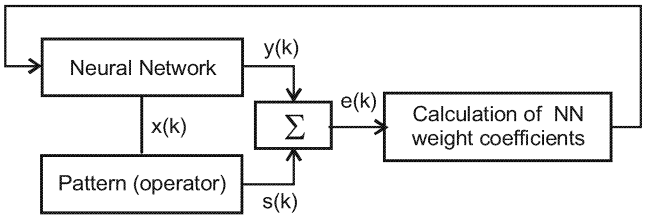


Fig. 3. Structure of learning the AI network under operator's (trainer's) supervisory

Realtionship between input and output is established by defined constraints. The weight coefficient w_i is calculated, for the k-th step of learning as:

$$w_{ij}(k+1) = w_{ij}(k) + \Delta w_{ij}(k) \tag{12}$$

where: i,j – number of joined neurons, k, $k+1$ – old and new number of w_i.

The process of learning may be tuned by the operator (figure 3.), each input vector x(k) is compared with stated vector s(t), thay are difined as:

$$x(k) = [x_1(k), x_2(k), \ldots x_n(k)]^T \tag{13}$$

$$s(k) = [s_1(k), s_2(k), \ldots s_m(k)]^T \tag{14}$$

The data learning NN are given for each group of input vector $x(k)$ and $s(k)$, for $k=1, 2,\ldots,p$, where p describe the learning pattern. For each pattern the error $e(k)$ is defined as:

$$e(k) = y(k) - s(k) \tag{15}$$

where: $y(k)$ is a current response of NN for input signal $x(k)$.

The pattern vector s_p is calculated a priori from data imported from SCADA system of PS for given states of committing PP and HP. It is possible to biuld in the pattern matrix to the simulator structure. The procedure of learning the NN, by the operator-expert, enables the possibility of minimization the error gained by the NN. For m neuron output and p patterns the error is described as:

$$E = \frac{1}{2}\sum_{k=1}^{p}\sum_{j=1}^{m} e_j^2(k) \tag{16}$$

Therefore the value of error of the calculated weight coefficients, defined upon the base of gradient of target function is:

$$\Delta w = -\eta \nabla E(w) \tag{17}$$

where: η – coefficient of learning effectiveness, defined as maximal different between adjacent step of learning the NN. After the end of given time period the cumulated value of this coefficient is known as stated value.

The structure of support the manager's decision consist of two expert systems. The first realises the diagnosis of current state vector of PS; the second gives some suggestion, about the probable decisions of PS manager, prepared upon the base of PS 's state recognition using the AI network.

For the PS manager it is very convenient if the decision is not precise, for example if it is fuzzy. That kind of decision is especially convenient in case of hydropower load scheduling, when the water stored in the reservoirs change theirs volume due to varied inflow and outflow.

RESULTS OF SIMULATION

The present principles of commitment include time and power gain without limits from the short and long term planning. Power plant framework optimisation is made using the linear or dynamic programming. The impact of manager's decision on natural environment is defined by different values of γ, coefficient which is substituting the increment of water consumption in accordance with increment of generations cost in thermal PP. The γ coefficient is responsible for allocation the hydro-power plant in the queue of load covering in accordance with disposed water volume. The κ coefficient is responsible for activation of the WT plants in accordance with power demand and disposed volume of water in the committing HP. Some results of simulation are shown in the fig. 4.

Allocation of peak load in thermal PP effects increase of emission CO_x, SO_x, NO_x and H_2O if the gas fuel is applying.

The assessed impact of the PS manager on the strategy of load of the hydro-power plant depends on the value of system equivalent price of water which is consumed in each hydro-power plant.

The value of EMV calculated for each PP and for both values of γ and κ shows that for low γ power plant TH2 is load as the first. If the value of γ is increasing the load is covered by power plant TH1. It means that in the first case the cost of emission of the combustion gases is lover than in the second one. Therefore the decision of PS manger effects the natural environment in two aspects: if the energy is generated by hydro-power plant the renewable energy is utilized and total amount of CO_x, SO_x, NO_x and H_2O is decreased. The total cost of generated energy changes from 15961,3 (γ=0,1and κ=0,8) to 6136,7 (γ=1,5 and κ=1,5) at the same level of production.

CONCLUSIONS

The described system simulates the range of possible decisions of the PS manager dedicated for teaching the manager in case to buy and sell the energy at rational level of risk.

The manager of a power system as the decision maker is

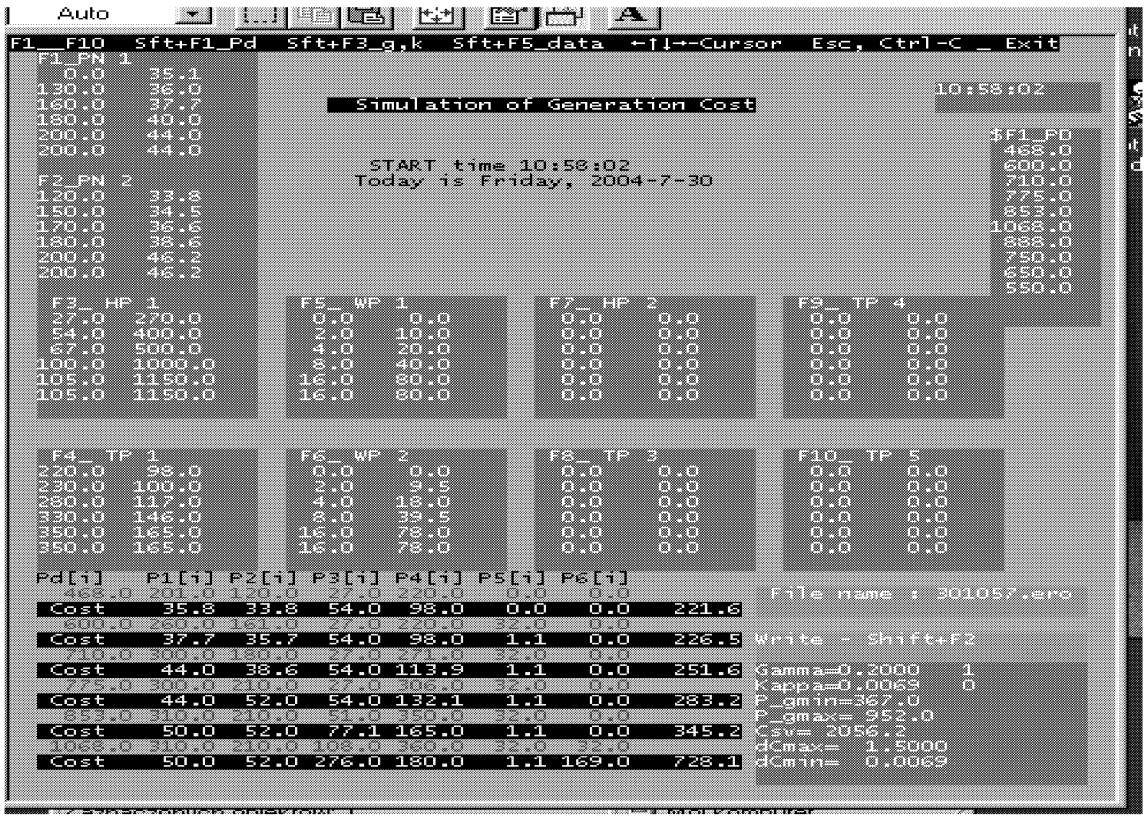


Fig. 4. Results of simulation the cost of energy generation

supported by the computer decision system with a special pre-calculated pay-off table for all possible scenarios of dispatch policy. The number of possible scenarios is limited with respect to power treated as a discrete value.

The aspects of validation of manager's decision take into account both components of analysis: the technical and economic effects. Considering the renewable sources of energy supplying the local markets (distributed generation) it is possible to formalize the impact of manager's decision on natural environment.

Using the described process of simulation it is possible to train the PS manager recognize the rules of optimal operation in the local energy market with regard to environment protection.

The advantage of proposed technique of teaching and training is tuning the simulator to the perception possibilities of the trained manager realized by using the factor of effectiveness of learning usually applied in the NN system.

REFERENCES

Abbasy N.H., Al-Hasawi W. M. 2003. Dynamic Modeling/Simulation and PerformanceAssessment of Kuwait Power Network Using PSS/E. *The 3rd IASTED International Conference on Power and Energy Systems, EuroPES 2003*, Marbella, Spain.

Baltierra A. E., Moitre D., Hernandez J. L. and Aromataris L. 1998. Simulation of an Optimal Economic Strategy of a Wholesale Competitive Electric Energy Market. In *Proc.of 10th European Simulation Symposium*, Nottingham, England. p. 255-259.

Chowdhury, Saifur Rahman 1990. A Review of Recent Advances in Economic Dispatch, *IEEE Trans. on PWRS*, Vol.5, No.4, Nov., pp.1248 -1259.

Das P., Haimes Y. Y. 1979. Multiobjective optimization in water quality and land management, *Water Resources Research*, vol.15, **6**, (1313-1322).

Malko J. 1997. Optimization of power generation structure on local market, *Rynek Energii* No. 5 (12), (in polish).

Sprague R. H. Jr. and Watson H.J. 1989. *Decision Support Systems Putting Theory into Practice*, Prentice Hall International Inc.

Sroczan E.and Urbaniak A. 2003. Simulation of manging the generation of ecologic energy in lokal energy market, In: *Simulation in Industry*, A. Verbraeck, V. Hlupic (eds.), SCS - European Publ. House, Delft - Erlangen - San Diego, (361 - 366)

Sroczan E. M., Urbaniak 2002. A Simulation of Variation the Power Load of Industrial Plant for Choice an Economy Rates of Energy. *14th European Simulation Symposium ESS 2002. simulation Industry, 23-26 Oct.,* Dresden Germany. p. 358-362

Sroczan E. 2000. Simulation of Power System Management using a Neural Networks and Genetic Algorithms. In *Simulation in industry* D. P.F. Moller (ed). Society for Computer Simulation International. Hamburg, Germany. p.279–281.

Sroczan E. 1995. Simulation of the Power System states applied for the optimization of energy management. *Proc.of 7th European Simulation Symposium ESS '95* Friedrich-Alexander University Erlangen - Nuremberg, Society for Computer Simulation International, Erlangen 1995 r. p .67-71.

Stoner J.A.F., Wankel C. 1986. *Management*, Prentice Hall Corp. England, New Jersey.

Yoshihara S., Shirai G., Fujita G., and Yokoyama R. 2003. Eigenvalue Analysis in TCSC Design for Oscillation Control. In: *The 3rd IASTED International Conference on Power and Energy Systems, EuroPES 2003*, Marbella, Spain.

PETRI NETS SIMULATION

TOLERANCE ANALYSIS APPROACH WITH INTERVAL CONSTRAINTED PETRI NETS

Collart Dutilleul S., Dhouibi H., Craye E.
LAIL, Ecole Centrale de Lille
Cité Scientifique, B.P. 48, France 59651Villeneuve d'Ascq
Tél. : (33-0) 03-20-33-54-04 ; Fax : (33-0) 03-20-33-54-18,
E-mail: collart_dutilleul@ig2i.fr , hedi.dhouibi@laposte.net, Etienne.Craye@ec-lille.fr,

Abstract: The existence of maximal durations drastically modifies the performance evaluation in Discrete Event Systems (DES). The same particularity may be found on systems where the associated constraints do not concern the time. The main contribution of this paper is to describe the behaviour of the Interval Constrained Petri Net (ICPN). This specification tool is designed in order to provide strong structural analysis. This tool is compared to the Abstract Marking High
Level Petri Net (AM-HLPN). The same mathematical abstraction is use in order to provide some constraints specifications. Nevertheless, the powerfulness of the AM-HLPN makes them difficult to
analyse. That's why we present these both tools as complementary ones. An industrial application
on a tobacco manufacturing plant is presented as an illustration.

Keywords: Petri-nets,
constraints, Time Petri-nets,
agile manufacturing,
performance evaluation

1. INTRODUCTION

Petri nets are known to be a well fitted tool to model synchronization and parallelism for Discrete Event Systems (DES). High Level Petri Nets (HLPN) have both ability of powerful modelling and integration of external parameters. Some works are leaded in order to get an ISO normalization (ISO/IEC 15909-1). The existence of maximal durations drastically modifies the performance evaluation in Discrete Event Systems (DES) and the classical formulas used for Cmax computing can not be used. A dedicated tool was introduced in 1996 in order to integrate the staying time constraints in a Petri Net model : the P-time Petri Nets (Khansa and all,1996a). This last tool offer both time constraints integration and strong analytical properties (Khansa and all,1996b),(Collart Dutilleul and all, 2002). From a specification point of view, they can be presented as a sub-class of mathematical abstraction of HLPN: with an abstracted Marking Petri Nets (AM-HLPN) (Collart Dutilleul and all, 2003a).
Moreover, the time is generally not the only parameter to take into account. There are a lot of industrial problems which concerns the control of parameters like positions, weights, etc… By the consequence, there is a need of functional generalisation of the mathematical properties of P-time PN. Therefore, Interval Constrained PN (ICPN) were presented (Collart Dutilleul and all, 2003b). At first, the extension of P-time PN properties to ICPN are Discussed. The main contribution of this paper is to propose an integrated approach in order to make a two-step schedule design. The first step is to specify a cyclic functioning for a system including several different kinds of parameters with AM-HLPN (Boussu and all, 1997), (Yim and all, 1996). Then, the cyclic functioning is studied from a disturbance point of view considering only one parameter. Finally, an illustration is given for a tobacco manufacturer, where the critical parameter is the weight.

2. HIGH LEVEL PETRI NETS

Autonomous Petri Nets are used in order to study logical properties like deadlock freeness. Nevertheless they are limited when a given quality has to be associated to tokens. Then timed extension like timed Petri Nets, t-Time Petri Nets and P-time Petri nets were introduced. On the other hand, the multiplication of different Petri Net extensions is not adequate with a strong and general theory. When dedicated tools are proposed for each case study, they will be a lack for general properties. Therefore, defining an innovative modelling tool with regard to HLPN is very interesting.

2.1. Basics (ISO/IEC 15909-1)

Definition:
A HLPN is a structure (S, T ,C ,F , Pre,Post,M_0) where:

- S is a finite set of places
- T is a finite set of transition. It is disjoint from S
- C is a non empty finite set of types
- F: $S \cup T \rightarrow C$ is a function used to type places and determine transition modes.
- Pre,Post : TRANS $\rightarrow$ µPLACE are the pre and post mappings with

$$TRANS=\{(t,m) \mid t\in T, m\in F(t)\} \quad (1)$$
$$PLACE=\{(s,g) \mid s\in S, g\in F(s)\}. \quad (2)$$

µPLACE is a multiset defined upon the set place

- $M_0 \in$ µPLACE is a multiset known as the initial marking of the net.

A marking is also a multiset: M∈µPLACE

Transition rule:
given a multiset of transition modes, T_μ, is enabled at a marking M, then a step may occur resulting in a new marking M' given by:

$$M'=M-Pre(T_\mu)+Post(T_\mu) \quad (3)$$

*2.2. AM-HLPN (*Boussu and all, 1997*)*

The Abstract Marked HLPN applied the concept of mathematical abstraction to marking of HLPN. In this case, it is no more possible to find an equivalent autonomous Petri Net model using unfolding. Nevertheless, this abstraction provides a very important increase of the modelling power. It is now possible to associate a parameter corresponding to the time to the tokens, like it was made in (Ghezzi and all, 1991). Then the time constraints are specified by propagating constraint of partial marking.

A partial marking is a 2-uple PM=(M,Cs) where :

- M is an application which associates to each token an expression as $k_0+k_1x_1+\ldots+k_nx_n$ where the k_i are integer constants and x_i are integer variables.
- Cs is a linear constraint containing the variables included in the M(s) expression.

Let θ be an assignment of the variables such that it satisfies C and that for each p θM(s) is a non negative integer.

Transition rule:
Let t be a transition and PM=(M,Cs) a partial marking. The constraint is called the firing constraint for the transition t. The transition is enabled for a partial marking PM if and only if the firing constraint can be satisfied.
The partial marking PM'=(M',Cs') resulting from the firing of t is defined by :

$$PM'=PM-Pre(t,s)+Pos(t,s) \quad (4)$$
$$Cs'=\Phi_t(PM) \quad (5)$$
$$\Phi_t(PM)=Cs \wedge \bigwedge_{s\in {}^\circ t} M(s) \geq Pos(s,t) \quad (6)$$

It is denoted: PM| t >PM'
Proposition:

Let t be a transition. PM and PM' are two different partial marking such as:

$$PM|t>PM'. \quad (7)$$

i) Each instance of PM' results from a firing of t from an instance of PM.

ii) Firing of a transition t from an instance of PM leads to an instance of PM'.

Considering the use of partial markings, it is possible to fire a transition without fixing the value of a given parameter. The associated constraints provide some specifications for this parameter value. By the consequence, AM-HLPN may be considered as a specification tool.

3. INTERVAL CONSTRAINED PN

For chemical process, for example, it is of importance to make sure that a ware will stay in a given state for a duration specified in a time window. Khansa W. proposed the P-time PN (Khansa and all, 1996a), (Khansa and all, 1996b). This extension of Petri nets can be modeled with AM-HLPN (Collart Dutilleul and all, 2003a). Moreover, these kinds of constraints may concern a parameter which is different from the time. The Interval Constrained PN were introduced (ICPN) in order to extend the functional range of applications of P-time PN (Collart Dutilleul and all, 2003b).

It may be a surprise, but the mathematical definition is not exactly the same. Actually, durations can not be negative, whereas it is completely different for a hoist position on a railroad.

Moreover, this new definition presents explicitly the place marking as a multi-set. Another aspect is the introduction of the transmitted parameter in the color of tokens as in (Ghezzi, and all,1991).

3.1 Definition :

An ICPN is a t-uple $<R, M, IS, D, Val, Val_0, X, X_0>$ where :

- R is a marked PN
- M is application associating token to places:

m is a vector indexed on the set of places P.

Let m(p) be a place marking.

Let V be a non empty set of rational variables.

Let μV be a multiset defined on V.

Let Val be an application: $M(P) \rightarrow \mu V$

$(k \in m(p), p) \rightarrow v \in \mu V$. Where k is a given token in p

- Val_0 corresponds to initial values associated to tokens.
- $IS : P \rightarrow Q \cup \{-\infty,+\infty\} \times Q \cup \{-\infty,+\infty\}$ defines the intervals associated to places.

$p_i \rightarrow IS_i = [a_i, b_i]$ avec $a_i \leq b_i$

- $D: M(p) \times P \rightarrow V$

$\forall i, 1 \leq i \leq n$, $n = Card(P)$

Let k be a token, $k \in m(p_i)$,

$$k \rightarrow q_i \mid a_i \leq q_i \leq b_i$$

- $X : V \rightarrow Q$

  $v \rightarrow u \in Q$
- X_0 is the vector of initial value of variables.

The significations of q and Val (k) are intentionally not fixed in order to provide a general model. Furthermore, a mathematical relationship between the q in different places is supposed to exist. Nevertheless it is not mathematically defined.

As an example with P-time PN there is the following relation:

$\forall q$, $dq/dt=1$, where t is the time.

With ICPN, X is not mathematically fixed.

3.2 State definition

A state E is a t-uple $< M, D, Val, X>$ where :

- M is the above defined application
- D is the application which associates to a token k in the place p_i a rational variable $q_i(k)$.

A token k of the place p_i can take a part in the validation of output transitions if:

$q_i(k) \in [a_i, b_i]$, where $[a_i, b_i]$ is the static interval associated to the place p_i.

This token k dies when: $q_i(k) > b_i$

In the following $q_i(k)$ are supposed to be monotonous. Nevertheless there is no mathematical definition of the evolution of the $q_i(k)$.

- Val is an application associating a formula based upon variables of V to each token. The effective values of variables take a part in the global state definition.

  - Val_0 is an application associating an initial value to each variable.
- X is an application which provides a value for each variable of V. Actually; X defines the real value of each q.

When X is not defined, there exists a way to make the model evolve. Furthermore, some mathematical properties may be outlined. It is the mathematical abstraction.

Definition: An abstraction on a set $\{x : A \bullet \psi\}$ may be interpreted as a set a value of the domain associated to the formula A. The A equation using x has to satisfy the constraint ψ.

The parameters defined with constrained variables will be specified, even though X is not defined, using the above definition:

$$A = Val_0(k) + \sum q_j, \quad (8)$$

the j index correspond to the visiting order of places by a given token

$x=\{ q_j \}$ **(9)**

$\psi=\prod (a_j \le q_i) \wedge (q_j \le b_i)$ **(10)**

The analytical conditions of a transition firing will be considered without fixing the final value of q variables. This approach is really useful to make a system specification.

3.3 validation of a transition

Lemme:
$\forall p_i \in {}^\circ t_l$, there exist at least Pré(p_i, tl) token in this place p_i such :
Let us denote I(p_i,t_l) a set of token which respects the conditions.
Let us denote X as a function of a set of variables $\{\rho_j\}_{j\in N}$. These variables corresponds to the coupling between the q_i values

i)When $\max_{k,i}(q_i(k)-a_i)<0$ then:

Let us define a set of values $\{\rho_{ji}, \rho_{jf}\}_{j\in N}$, where N is the set of integer numbers.
ρ_{ji} is the initial current value of the variable ρ_j.
ρ_{jf} is the final value such that:

$$\begin{aligned} &abs(\rho_{jf*}-\rho_{ji}) < abs(\rho_{jf}-\rho_{ji}) \\ \Rightarrow \exists k \in I(p_i,t_l) \mid\ & ai - Xm(k)_{\{\rho_{ji}\}_{j\in N}} < 0 \end{aligned} \quad \textbf{(11)}$$

Then the following condition has to be fulfilled:

$$\begin{cases} bn - Xn(k^*)_{\{\rho_{ji}\}_{j\in N}} = \min_{k,i}(b_i - q_i(k)) \\ bn - Xn(k^*)_{\{\rho_{ji}\}_{j\in N}} \ge 0 \end{cases} \quad \textbf{(12)}$$

with k=1, 2, ..., Pré(p_i,t_l) and $[a_i, b_i]$ is the static interval associated to the place p_i.

ii)When $\max_k(a_i-q_i(k))<0$ then:

$$\min(b_i - q_i(k)) \ge 0 \quad \textbf{(13)}$$

iii) Furthermore, there is no token r such:
$r \notin (p_i,t_l)$, $r \in M(p_z)$
When $\max_{k,i}(q_i(k)-a_i)<0$ then:

$$bz - Xz(r)_{\{\rho_{ji}\}_{j\in N}} \ge 0 \quad \textbf{(14)}$$

When $\max_{k,i}(q_i(k)-a_i)<0$ then:

$$q_z - b_z > 0 \quad \textbf{(15)}$$

Definition
A transition t_l is validated from a state E (M, D,Val) if and only if :

1- it is validated in the sense of autonomous RdP :
$\forall p_i \in {}^\circ t_l$, $m(p_i) \ge$ Pré(p_i,t_l) ;

2- The conditions of lemme1 are fulfilled

3- $\forall p_i \in {}^\circ t_l$,
$\exists I(p_i,tl) | \forall n_i \in I(p_i,tl) |$ val(n_i) + ($q_i(n_i)$)=K, **(16)**
where K is a constant value.

The above conditions are building upon the monotonous assumption of X. Without this assumption the use of validity interval would be more difficult.
When the application X is not defined, the transition firing modifies the constraints established on variables (the q variables). A logical product is made with the new constraints and the ones which already exist. Actually the above conditions can not be checked.

3.4 Computing the next step

There are tow different ways of reaching a state from a given one. The first solution is to use the evolution of associated variables. The second one is the transition firings. The following two definitions correspond to these two possibilities of evolution.

Definition:
The state E'(M', D', Val') can be reached from another one E (M, D, Val) using the evolution of associated variables if and only if:

1- M'=M,

2- $\forall j \in m(p_i)$:
$q'_i(j) = q_i(j) + \Delta q_i(j)$, $a_i \le q'_i \le b_i$, **(17)**
Where $[a_i , b_i]$ is the static interval of the place p_i .

the possibility of reaching $q'_i(j)$ depends generally on the coupling with other q evolutions. This particular aspect is not presented here.

Definition:
The state E'(M', D', Val') can be reached from another one E(M, D, Val) by firing a transition t_i if and only if :

1- t_i is validated from E,
2- $\forall p \in P$,
$M'(p) = M(p) - Pré(p, t_i) + Post(p, t_i)$, **(18)**

3- Tokens which do not move keep the same associated parameter in E et E'. The moving or the created tokens takes the zero value for the local q associated to their new place. The value assign to the token k' by the application Val is:

$Val(k') = Val(k) + q_i(k)$, **(19)**

k is a token which where in the input place of t_i and which was consumed by firing t_i.

The above firing rules allow computing the states and the accessibility relationship between us.
The set of valid firing sequences from the initial state characterizes the net behaviour.

3.4 P-time PN: a sub-class of ICPN

Actually, P-time PN may be defined as a sub-class of ICPN. It use the following restrictions:
The arrival set off IS application is restricted from $Q \cup \{-\infty,+\infty\} \times Q \cup \{-\infty,+\infty\}$ to $Q^+ \cup +\infty \times Q^+ \cup +\infty$.
The signification of the parameter associated to the tokens is the age. This restriction respects the ICPN definition.
The X application which provides the definition of parameter's evolution is strictly fixed, because it corresponds to the linear time evolution. Let us remark that this function match the monotonous criteria.

ICPN may be seen as a functional abstraction of P-time PN. The time is only a particular value in the domain of possible interpretations of parameters.

Theorem: structural mathematical properties of P-time PN can be extended to ICPN if and only if:

$\forall i |\ 1 \leq i \leq n$, n= Card (P), mini $(q_i) \neq -\infty$

Of course, the properties extension implies that the structure and the marking restriction are the same.

Proof: ICPN instances have all the same mathematical model. The only specificity of P-time PN is the restriction of the token associated parameter to positive rational numbers and the definition of the X application.

Consequently, any demonstration which do not use an assumption on X definition and which only use the marking and the net structure will be valid for any instance of ICPN.

There is only one condition to be fulfilled: the minimal values for the static intervals associated to places must not be equal to minus infinite. Actually, if this minimal value is negative and finite, a simple change of the origin on the axe of parameter values will shift the arrival set of IS intervals into positive values.

Nevertheless, the physical interpretation of the same property in each instance of model will depend on the physical meaning of parameters which are associated to the tokens.

4. A CIGARETTE FACTORY ANALYSIS

tobacco being solds in volume, the manufacturers of the cigarettes have a big interest to reserve to their machines tobacco most capable to permit a matter economy as important as possible, without altering the characters of the tasting of the product. they will try therefore to prepare the maximum of cigarettes with the minimum of tobacco while being careful to respect the norms of quality. Indeed, to get a good quality and to minimize the made cigarette cost, these must have an optimum weight understood in one definite interval. So the weight constraint is critical for cigarettes manufacturing. Actually, it modifies the quality and the cost of the cigarettes. When it is outside of the validity range, the production has to be rejected.

Considering the quality, a too weighty cigarette is difficult to smoke, but when it is not enough the consumer feel stolen. Moreover, the cigarettes extremity is not solid.

For a batch of cigarettes, the weight parameter is a function of three parameters.

The diameter (**m**), the length (**l**) of the cigarette and the compactness (density) of the tobacco (**c**).

Considering a cigarette as a cylinder of diameter (m) with length (l), the following relationship may be written:

$$P = km^2.l.c \qquad \text{where:} \qquad \textbf{(20)}$$

m is the diameter of the cigarettes in mm. There is a validity range fro this parameter : $\mathbf{m \in [m_{min}, m_{max}]}$.
An out of range value of the module will produce a machine blocking. In this case the workshop is stopped and all the machines have to be emptied.

l is the length in mm. This parameter may block the plant at the packing stage: $\mathbf{l \in [l_{min}, l_{max}]}$.

c is the compactness or density in g/mm^3. There are quality specifications on this parameter. A cigarette with a too high compactness is difficult to smock:
$\mathbf{c \in [c_{min}, c_{max}]}$.

k: is a constant value ($\Pi/4$)

4.1 Scheduling the workshop

From a specification point of view, it was shown that ICPN can be simulated with AMHLPN. This modelling tool may be found in the literature with the name Hypernet (Yim and all, 1996). The Hypernets were proposed an associated solving tool named Minos which based upon a constraint programming language (Prolog) (Yim and all, 1995). Then it is possible to make the scheduling task from a graphic modelling tool.
The tool is able to model time window constraints. Moreover, other kind of constraints may be integrated. In the presented case study, a nonlinear equation including four parameters has to be handled. A 4–uple of rational variables is associated to the tokens.
Then, the associated solver may be used in order to provide a cyclic scheduling with some reference parameter value. Nevertheless, it will not be presented here as it is not the main subject of this paper.

Finally, when the schedule is established, the model of the system can be reduced to a Strongly Connected Event Graph. The P-time PN may be used in order to check some demanded properties. At this stage, the modelling powerfulness of the Hypernets or AM-HLPN is no more appropriate; because the objective is now to find some mathematical proves based upon structural properties.

4.2 Property checking

Obviously, the variation of one of these parameters provides a variation of the weight. The aim of the controller if to maintain the weight specification by changing the setting parameter, whereas they have to remain in a validity interval:

$$P_{\min} \le k.m^2 l.c \le P_{\max} \qquad \textbf{(21)}$$

Linearization in the region of a setting point

The variation of a parameter are always very small in front of is setting value. By the consequence, the above relation may be approximated by the following one doing a first order linearization:

$$\Delta P \in [\Delta P_{\min}, \Delta P_{\max}]$$

$$\Delta P \le 2klcm\Delta m + km^2 c\Delta l + km^2 l\Delta c \qquad \textbf{(22)}$$

The parameter of this last relation (22) may be integrated in the Petri net model of the figure 1.

The manufacturing process corresponding to the place Pc and Pm (compactness and diameter) is continuous. It is considered as discrete in this analysis part because the measures are made all the 20 cigarettes. K_1=klm², K_2=2klm , K_3=kcm²

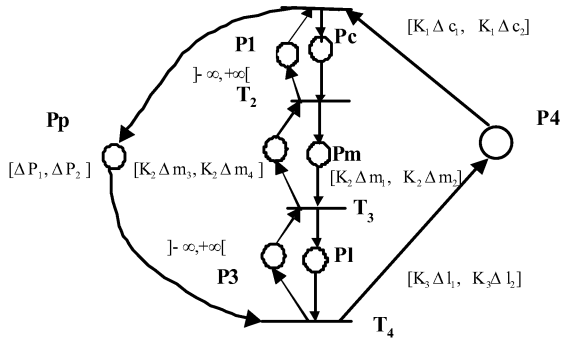


Figure 1. RdP with the weight constraints

Uncontrollable variation may occur in the compactness, because it depends on the quality of the tobacco. There is one more constraints concerning the variation of the weight all the 60 cigarettes (figure 2).

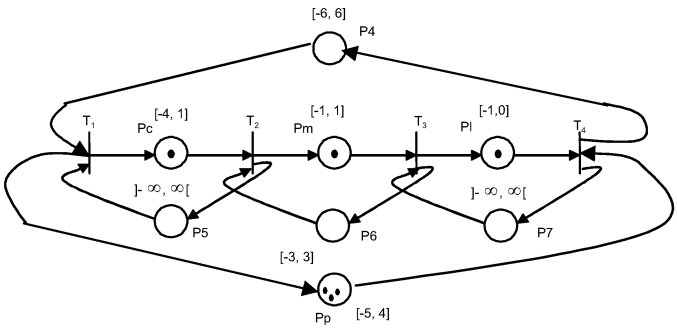


Figure 2. *ICPN corresponding to the workshop*

There are five circuits in this PN graph:

$$\gamma_1 = (T_1, P_c, T_2, P_5)$$

$$\gamma_2 = (T_2, P_m, T_3, P_6)$$

$$\gamma_3 = (T_3, P_l, T_4, P_7)$$

$$\gamma_4 = (T_1, P_c, T_2, P_m, T_3, P_l, T_4, P_4)$$

$$\gamma_5 = (T_1, P_p, T_4, P_4)$$

There is one couple parallelism and synchronization with two parallel paths:

$$((T_1, P_c, T_2, P_m, T_3, P_l, T_4), (T_1, P_p, T_4))$$

It is easy to see that the fifth elementary circuit is a linear combination of the four others and the couple parallelism/synchronization.
On the figure 2, the zero values of the intervals correspond to the reference setting point values. Actually, this reference setting point is applied such a way that the cycle weight is equal to zero.
Then, the robustness towards a compactness variation will be studied.

*Using a theorem of P-time PN (*Collart Dutilleul and all, 1999)

Definition: every structure of the net like elementary circuits and paths between couples (synchronization, parallelism) will be called critical structure.
SC_i corresponds to the critical structure which index is i.
nc_i is the number of tokens in SC_i.
The constraints model can be derived from critical structure containing exactly one disengaged hoist travel place.
This place will be denoted p_r and the corresponding critical structure SC_r.
Let us denote Ct_i the settings of staying times where C_i is the corresponding cycle time.
$At_{i,Ctj}$=($q_{i,Ctj}$ -$a_{i,Ctj}$). $At_{i,Ctj}$ is the disposability margin of a resource or a constraint expressed by the place p_i. This margin is available for the setting Ct_j.
SC(ε) is the set of critical structure which are elementary circuits when ε=+1, or couples of path between parallelism and synchronization when ε=−1.

Theorem:
Let us denote Δd the variation of a given operation. Let us consider a P-time SCEG which can be reduced to a set of critical structures, a sufficient condition to keep on functioning any temporal constraint violation is:

$$\Delta d \in [\min(0, \varepsilon.\min_{r\in SC(\varepsilon)}.(At_{r,Ctl}/nc_r)), \max(0, \varepsilon.\min_{r\in SC(\varepsilon)}.(At_{r,Ctl}/nc_r))] \qquad \textbf{(23)}$$

The above theorem allows us to prove weather the current setting is able to face the uncontrolled variation of the

tobacco compactness. The reference cyclic state corresponds to a zero value assign for each q.
Let us consider a compactness decreasing. The maximal valid compactness decreasing is equal to 4.
Let us compute all the local static margins:
$At_1 = -\infty$
$At_2 = -3$
$At_3 = -\infty$
$At_4 = -5/3$, where $nc_4 = 3$
$At_5 = -6/3$, where $nc_5 = 3$ and SC_5 is the couple parallelism /synchronization

By the consequence, for a compactness decreasing, the static robustness is such:

$$\varepsilon.\min_{r \in SC(\varepsilon)}.(At_{r,Ctl}/nc_r)=At_4 \quad \Rightarrow \quad \Delta d \in [0, -5/3] \qquad \textbf{(24)}$$

Then, this last value can be used to set a watchdog on the system, because when the compactness variation exceeds -5/3, other parameters have to be changed in order to keep the global weight constraint on the cigarettes.
Some properties concerning the transient mode from a setting point to another (Collart Dutilleul and all, 2002). It may be interesting to use them in this last case.

5. CONCLUSION

The ICPN detailed functioning was presented is this paper considering the HLPN reference. Then, the P-time PN properties extension to its functional abstraction the ICPN was discussed. Then some sufficient conditions were provided. ICPN, like P-time PN, may be used in a global approach based upon AM-HLPN (Collart Dutilleul and all, 2003). The high level net is used in order to specify a general design, whereas the ICPN will provide their strong structural properties to prove some analytical properties.
On a real instance of a tobacco manufacturing plant, some basic properties were illustrated. The performance evaluation can be applied and an internal robustness proof was provided.
There is still a lot of thing to do with the AM-HLPN step with non linear constraint on several parameters.
Actually, the presented results concerning robustness let a wide research area. Actually, the existing properties ensue only from staying time problem topologies.

REFERENCES

[Boussu and all]: Boussu F., Happiette M., Lefort A., Rabenasolo B., Yim P. "Modelling and Simulation of the Textile Channel by Hypernets" Journal of the Textile Institute, Vol.88, n°2, pp146-157, 1997.

[Collart Dutilleul and all,1999]: Collart Dutilleul S., Denat J.P., " Integration of the chemist expertise in electroplating line automation", ECC'99, Karlsruhe, Germany.

[Collart Dutilleul and all, 2002]: Collart Dutilleul S. , Denat J.P, Chetouane F. 2002 " External Robust Control of Electroplating Lines " IEEE Conference- Hammamet October 2002, Tunisia.

[Collart Dutilleul and all, 2003a]: Collart Dutilleul S., P.Yim., "Time window specification and validation with Petri nets", ETFA IEEE conference,2003, Lisbone, Portugal.

[Collart Dutilleul and all, 2003b]: Collart Dutilleul S., H. Dhouibi, E.Craye., "Internal Robustness of Discret Event System with interval constraints in repetitive functioning mode. ", ACS'2003 conference, Międzyzdroje, Poland.

[Ghezzi and all,1991]: Ghezzi C., D. Mandrioli, S. Morasca, and Pezzè, " A unified high level petri net formalism for time-critical systems" IEEE Transaction on Software Engineering, SE-17(2), 1991.

[ISO/IEC 15909-1]: International Standard Ballot of ISO/IEC 15909-1 (High-level Petri Nets)

[Khansa and all,1996a]: Khansa W. P. Aygalinc, J.P. Denat. , " Structural analysis of p Time Petri Nets", *CESA '96* , Lille.

[Khansa and all,1996b]: Khansa W., Denat J.P., Collart Dutilleul S.,1996, "P-time Petri Nets for Manufacturing Systems", Proc. of the IEE International Workshop on Discrete Event Systems (WODES'96), Edinburgh, Scotland, August 96, pp. 94-102.

[Yim and all, 1995]: Yim P., Lefort A., Hebrard A. "MINOS: a constraints based tool for the hypernets" 7th IEEE Int. Conf. on "Tools with Artificial Intelligence", ICTAI'95, Washington (U.S.A.), November 1995, pp 334-337.

[Yim and all, 1996]: Yim P., A. Lefort, and A. Hebrard " System Modelling with Hypernets" ETFA'96 IEEE Conference, Paris , October 1996, pp 37-47.

MULTILAYER DECOMPOSITION OF STOCHASTIC PETRI NETS AND ALGEBRAIC SPECIFICATIONS

Nabila SALMI
Malika IOUALALEN
Université des Sciences et Technologie de Houari Boumediene
Faculté d'Informatique & d'Electronique
Département informatique
BP 32. El-Alia. Bab-Ezzouar. Alger. Algérie.
http://www.usthb.dz

KEYWORDS
Stochastic Petri Nets, Performance evaluation, States Space explosion, decomposition, algebraic expression, synchronous composition, asynchronous composition.

ABSTRACT

Numerical analysis with stochastic Petri nets is difficult when dealing with large and parallel systems, because off states space explosion. In order to avoid this problem, we propose a decomposition technique which splits the system into subsystems and allows t0 write the analysed system as an algebraic expression combining subnets of the original Petri net with algebraic operators. This expression will be useful to reduce the complexity of numerical analysis of Stochastic Petri Nets, by analysing subnets of the original net instead of analysing the whole net.

I. INTRODUCTION

Stochastic Petri Nets (SPN) have been widely used in the last years for performance analysis of parallel and complex systems, especially in telecommunications and production systems. They are considered as a powerful model which expresses the most important characteristics and permits, thanks to its probabilistic nature, to perform a numerical performance evaluation. In fact, they have proved their efficiency with analytical results, inspired from stochastic processes, especially Markov chains. However, the numerical analysis often leads to the well known problem of **states space explosion,** when the SPN is large. This problem makes the analysis very difficult, and even impossible.
To avoid this problem, a lot of work has been proposed. This work can be classified in three classes of techniques:
1-Model transformation, composition and decomposition, and state aggregation methods (Dutheillet and Haddad 1989), (Dutheillet 1992), (Haddad, Ilié, Taghelit and Zouari 1995), (Ioualalen,Boukala and Aissani 1998).
2-Techniques for the computation of product form solution, inspired from queuing system analysis (Donatelli and Sereno 1991), (Florin and Natkin 1989).
3-Techniques computing bounds for performance indicators (Truffet 1995), (Zuberek 1989).

Even though the use of such methods reduces the impact of states space explosion, the size of nowadays systems increases more and more with the advent of new multimedia applications and the use of all kind of networks (LAN, WAN, Wireless, adhoc, …). Thus, the need for new approaches still exists.
Among these techniques, there were many attempts to identify subnets composing an SPN, in such a way that we analyse the subnets, and solve the stochastic process associated with the original SPN, without generating the complete Markov Chain (Donatelli and Sereno 1991), (Plateau and Fourneau 1991), (Buchholz 1992), (Buchholz 1997), (Haddad and Moreaux 1997). In this work, identification of subnets is done manually by the user, by searching two possible dependencies between components :

- Components can communicate by places through the exchange of tokens: this represents asynchronous communications between subsystems.
- Components can communicate by transitions which synchronize several activities: this represents synchronous communications between subsystems.

To solve the SPN decomposed into several subnets, these attempts exploit kronecker algebra (Davio 1981) to express the SPN global solution as a function of subnets solutions, without need to store the infinitesimal generator of the complete Markov chain.
Extending this concept, we present in this paper, a new decomposition technique of stochastic Petri nets, which takes advantage from the synchronous and asynchronous composition of subnets. The aim of this approach is to identify subnets composing the net under analysis, and deduce relations between these subnets, giving at last an algebraic expression of the SPN, composed by identified subnets linked by operators which will be defined.
This decomposition technique will permit to simplify the performance analysis of SPN, by analysing the components (subnets), and then computing the global SPN solution, on the basis of the algebraic expression found and kronecker algebra. As we focus in this work on the decomposition technique, the computation of the global solution thanks to the algebraic expression will be clarified in another work.
This paper will be organised as follows: we first recall SPN concepts and also some graph theory definitions. Then, we introduce special subnets which help us in

identifying subnets in an SPN. After that, we define a set of operators and so an SPN algebra, used to construct the algebraic expression of an SPN. Finally, we give the decomposition approach from which we obtain identified subnets and the algebraic expression binding them. An example illustrates the proposed algorithm.

2. PRELIMINARY CONCEPTS

Definitions below introduce Stochastic Petri Nets (SPN) and Generalized SPN as a basic model in our technique.

Def 1: A Stochastic Petri Net (SPN) is a couple S=<N,W> where:

- N is a marked Petri net, N = (P, T, Pre, Post, Inh, M_0) :
  - P, T are 2 finite disjoint sets called places and transitions.
  - Pre, Post and Inh : PxT → Bag(P) are arc functions :
    - Pre, Post, Inh are resp. backward incidence, forward incidence and inhibition functions,
    - Bag(P) is the set of multisets on P: Bag(P)={x=$(x(p))_{p\in P}$ / ∀p∈P, x(p)∈ IN}.
  - M_0 is a function from P to N called the initial marking.
- W = (λ_1,.....,λ_n) is the vector of transition rates.

The set of the input places (respectively output places) of a transition is: $^{\cdot}t$ = {p∈ P / Pre (p, t) >0 }

Respectively $t^{\cdot}$ = {p∈ P / Post (p, t)>0 }

Def 2: A Generalized Stochastic Petri Net (GSPN) is a couple S = <N, θ> where:

- N is a marked stochastic Petri net, N = (P, T, Pre, Post, Inh, M_0) where T=$T_E \cup T_I$:
  - T_E : is the set of **timed** transitions.
  - T_I : is the set of **immediate** transitions.
- θ : Tx Bag(P)→$|R^+$ associates :
  - a firing rate to a timed transition.
  - A weight or priority to an immediate transition.

Before introducing sorts of subnets we rely on in our approach, we recall the definition of a Petri net based on the graph theory, and define other inherent concepts.

Def 4: A Stochastic Petri Net (SPN) S = <N, W> is a bipartite oriented graph G_N=(X,E), where :

- X : is a set none empty of vertices of G : X= P ∪ T.
- E : is a set of arcs between the nodes of G.

We define two functions I and T / I, T: E → X called respectively **initial end and final end** of G.

Def 5: Degree of a node

The **degree** of a node X, denoted **d(X)** is the number of incoming and outgoing arcs of X.

- The **outgoing (output) degree $d^+(x)$** is the number of outgoing arcs of X : $d^+(x)$ =| {e ∈ E / I(e)=x } |
- The **incoming (input) degree $d^-(x)$** is the number of incoming arcs of X : $d^-(x)$ =| {e ∈ E / T(e)=x } |

Def 6: Successor and predecessor

Let G be a graph of a Petri net/ G=(X, E), X= P ∪ T.

A node y (place or transition) is a **successor** of another node x (resp.x is **predecessor** of y) if : ∃ e∈ E / e=(x, y).

$X^{\bullet}$ = { y∈X / e=(x, y)∈ E} the set of successors of X.

$^{\bullet}X$ = {y∈ X / e=(y, x) ∈ E} the set of predecessors of X.

Def 7: Source and well

Let G be a graph of a Petri net/ G=(X,E), X= P∪T. A node x is called **Source** (resp.**Well**) iff : $d^-(x)$ =0 (resp.$d^+(x)$=0).

Definition 8: Path

A **Path** in an oriented graph G=(X, E) is an alternate continuation of nodes and arcs $\gamma=x_0e_1x_1e_2\ldots x_{k-1}e_kx_k$/ i∈ [1,k], e_i=(x_{i-1},x_i). We say: γ is a path with **length k** from x_0 to x_k .The path γ is **elementary** if all its nodes are distinct.

Def 9: Circuit

A **Circuit** is a closed path (k>0 and $x_0=x_k$).

Def 10: Subnet Interface

Let G be a graph of a Petri net/ G=(X,E), X= P∪T. Let G' be a subnet of G so that G'=(X', E'). We call **Interface** of G' the set of nodes of G' linked to the outside of G', denoted **Intf(G')**, and which is defined as :

Intf(G')={x' ∈ X'/ x ∈ X-X', (x', x) ∈ E or (x, x') ∈ E}

The incoming (Input) interface, **$Intf^-(G')$**, is the set of nodes of G', linked to the outside of G', by incoming arcs.

The outgoing (Output) interface, **$Intf^+(G')$**, is the set of nodes of G', linked to the inside of G', by outgoing arcs.

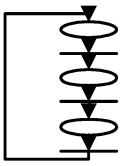
Next, we define different subnets sorts we use later.

3. DEFINITION OF TYPES OF SUBNETS

Def11: None Choice/Synchronization Petri Net (NCSN)

An SPN S=<N,W>, with N= (P, T, Pre, Post, Inh, M_0) is said **None Choice/Synchronization Petri Net (NCSN)** if and only if : ∀ x ∈ X, $d^+(x)$= $d^-(x)$ =1

Example 1 :



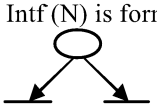
[Fig.1 : NCSN]

Def 12: Petri Net SN_PnT

An SPN S = <N, W>, with N = (P, T, Pre, Post, Inh, M_0) is said **SN_PnT** if and only if:

- $|Intf^+(N)|$ = 1 / $Intf^+(N)$ is formed of one place a, and
- $|Intf^-(N)|$ = n / $Intf^-(N)$ is formed of n transitions.

Example 2 :



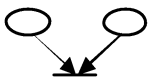
[Fig.2 : SN_P2T]

Def 13: Petri Net SN_nPT

An SPN S = <N,W>, with N = (P, T, Pre, Post, Inh, M_0) is said **SN_nPT** if and only if :

- $|Intf^+(N)|$ = n / $Intf^+(N)$ is formed of n places, and
- $|Intf^-(N)|$ = 1 / $Intf^-(N)$ is formed of one transition.

Example 3 :



[Fig.3 : SN_2PT]

Def 14: Petri Net SN_nTP

An SPN S = <N, W>, with N = (P, T, Pre, Post, Inh, M_0) is said **SN_nTP** if and only if :

- $|Intf^+(N)|$=n / $Intf^+(N)$ is formed of n transitions, and
- $|Intf^-(N)|$=1 / $Intf^-(N)$ is formed of one place.

Example 4 :



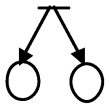
[Fig.4 : SN_2TP]

Def 15: Petri Net SN_TnP

An SPN S = <N, W>, with N = (P, T, Pre, Post, Inh, M_0) is said **SN_TnP** if and only if :

- $|Intf^+(N)|$ = 1 / $Intf^+(N)$ is formed of one transition, and
- $|Intf^-(N)|$ = n / $Intf^-(N)$ is formed of n places.

Example 5 :



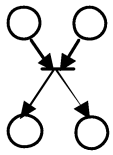
[Fig.5 : SN_T2P]

Def 16: Petri Net SN_nPTmP

An SPN S = <N, W>, with N = (P, T, Pre, Post, Inh, M_0) is said **SN_nPTmP** if and only if :

- $|Intf^+(N)|$ = n / $Intf^+(N)$ is formed of n places, and
- |Intf(N)| = m / Intf(N) is formed of m places.

Example 6 :



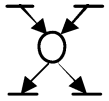
[Fig.6 : SN_2PT2P]

Def 17: Petri Net SN_nTPmT

An SPN S = <N,W>, with N = (P, T, Pre , Post, Inh, M_0) is said **SN_nTPmT** if and only if :

- $|Intf^+(N)|$ = n / $Intf^+(N)$ is formed of n transitions, and
- |Intf(N)| = m / Intf(N) is formed of m transitions.

Example 7 :



[Fig.7 : SN_2TP2T]

How these sorts are used in our decomposition technique? The decomposition technique consists of searching in the SPN if there is a subnet whose type corresponds to one of those defined above, and progressively identify relations between these subnets, in order to construct an algebraic expression of the whole SPN. To do that, we need to an SPN algebra, and subnets operators. This is defined below.

4. STOCHASTIC PETRI NET ALGEBRA

In this part, we introduce an algebra which permits to compose SPNs by using operators. For the generality of the case, this algebra is defined on the set of GSPNs.

4.1. Definition Of GSPN Algebra

Def 18: GSPN Algebra

A GSPN **algebra** is a triplet <S, $\sum$, E> where :

- S : is the set of GSPN N=*(P, T,Pre,Post,Inh,Pri, M_0, θ)*,
- $\sum$: is a set of operators defined on S.

4.2. Operators Of GSPN Algebra

The set $\sum$ of operators is defined generally between 2 GSPNs. For the following, let us consider N_1, N_2 so that :

$N_1 = (P_1, T_1, Pre_1, Post_1, Inh_1, Pri_1, M_{01}, \theta_1)$,

$N_2 = (P_2, T_2, Pre_2, Post_2, Inh_2, Pri_2, M_{02}, \theta_2)$.

4.2.1. Choice operator

The choice between two Petri subnets models two possible behaviours, among which one only must be chosen, on the basis of the selection of an activity among two. The activity selected initiates the behaviour of the Petri net. In this case, graphically, each of the two subnets is connected to the other part of the Petri net by a transition. The two transitions associated with the two subnets are said **in structural conflict**. This definition is given below :

Def 19: Structural Conflict (Moreaux 1996)

A transition t is in **structural conflict** with a transition t' (denoted **t SC t'**) if and only if :

($^\bullet t \cap {}^\bullet t' \neq \varnothing$) ou ($t^\bullet \cap {}^\circ t' \neq \varnothing$) with ${}^\circ t' = \{p \in P / Inh(p, t) \neq 0\}$

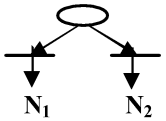
Def 20: Choice operator +

N_1+N_2 is a GSPN $N=(P_1 \cup P_2, T_1 \cup T_2, Pre, Post, Inh, Pri, M_0, \theta)$

- Pre : $(P_1 \cup P_2, T_1 \cup T_2) \rightarrow |N$ so that :
  $Pre(p,t) = pre_1(p,t)$ if $p \in P_1$ and $t \in T_1$
  $Pre(p,t) = pre_2(p,t)$ if $p \in P_2$ and $t \in T_2$, and 0 otherwise.
- Post : $(P_1 \cup P_2, T_1 \cup T_2) \rightarrow |N$ so that:
  $Post(p,t) = post_1(p,t)$ if $p \in P_1$ and $t \in T_1$
  $Post(p,t) = post_2(p,t)$ if $p \in P_2$ and $t \in T_2$, and 0 otherwise.
- Inh : $(P_1 \cup P_2, T_1 \cup T_2) \rightarrow |N$ so that:
  $Inh(p,t) = Inh_1(p,t)$ if $p \in P_1$ and $t \in T_1$
  $Inh(p,t) = Inh_2(p,t)$ if $p \in P_2$ and $t \in T_2$, and 0 otherwise.
- Pri : $(T_1 \cup T_2) \rightarrow |N$ so that: $Pri(t) = pri_1(t)$ if $t \in T_1$,
  $Pri(t) = pri_2(t)$ if $t \in T_2$, $Pri(t) = 0$ otherwise.
- θ : $(T_1 \cup T_2) \rightarrow |N$ so that: $\theta(t) = \theta_1(t)$ if $t \in T_1$,
  $\theta(t) = \theta_2(t)$ if $t \in T_2$, $\theta(t) = 0$ otherwise.

Semantics of the choice operator

The choice between two Petri subnets is supported by two transitions which are in structural conflict, and having a common place (common resource) as preconditions.



In this case, a race policy is used to decide which behaviour to choose, when several activities are possible simultaneously (i.e. corresponding transitions enabled). As a probabilistic distribution is associated with the firing duration of each transition, the SPN to be chosen is that connected to the transition having ***the smallest sampled duration*** of firing.

4.2.2. Parallelism Operator ||

Parallelism between two Petri subnets models two behaviours which can be carried out simultaneously.

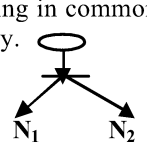
Def 21: Parallelism operator

$N_1||N_2$ is a GSPN $N=(P_1 \cup P_2, T_1 \cup T_2, Pre, Post, Inh, Pri, M_0, \theta)$:

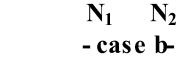
- Pre : $(P_1 \cup P_2, T_1 \cup T_2) \rightarrow |N$ so that :
  $Pre(p,t) = pre_1(p,t)$ if $p \in P_1$ and $t \in T_1$
  $Pre(p,t) = pre_2(p,t)$ if $p \in P_2$ and $t \in T_2$, and 0 otherwise.
- Post : $(P_1 \cup P_2, T_1 \cup T_2) \rightarrow |N$ so that:
  $Post(p,t) = post_1(p,t)$ if $p \in P_1$ and $t \in T_1$
  $Post(p,t) = post_2(p,t)$ if $p \in P_2$ and $t \in T_2$, and 0 otherwise.
- Inh : $(P_1 \cup P_2, T_1 \cup T_2) \rightarrow |N$ so that:
  $Inh(p,t) = Inh_1(p,t)$ if $p \in P_1$ and $t \in T_1$
  $Inh(p,t) = Inh_2(p,t)$ if $p \in P_2$ and $t \in T_2$, and 0 otherwise.
- Pri : $(T_1 \cup T_2) \rightarrow |N$ so that: $Pri(t) = pri_1(t)$ if $t \in T_1$,
  $Pri(t) = pri_2(t)$ if $t \in T_2$, $Pri(t) = 0$ otherwise.
- θ : $(T_1 \cup T_2) \rightarrow |N$ so that: $\theta(t) = \theta_1(t)$ if $t \in T_1$,
  $\theta(t) = \theta_2(t)$ if $t \in T_2$, $\theta(t) = 0$ otherwise.

Semantics of the parallelism operator

Parallelism between two Petri subnets models two processes which can be carried out simultaneously. Therefore, the behaviour of each subnet is independent of the other one. These subnets can be related by a common transition (case a), or can be fully independent, that is they do not have anything in common (case b). They evolve in an independent way.



- case a -



- case b-

4.2.3. Synchronous composition operator ▶◀

The synchronous composition of GSPN models synchronization of events associated with the subsystems

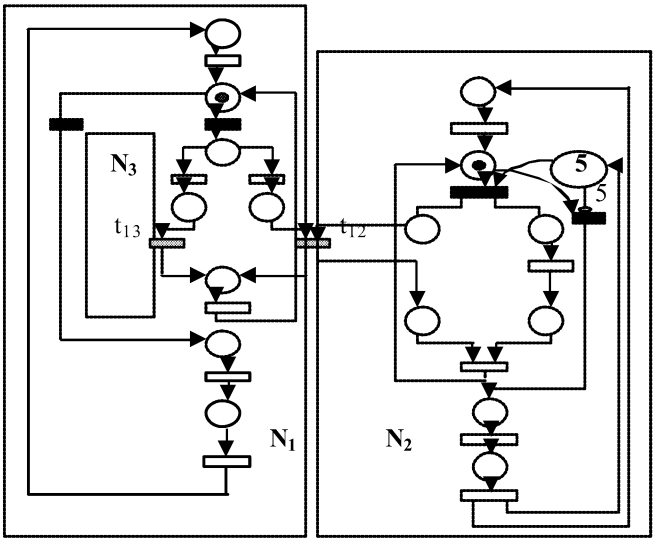
represented by these GSPN (Example: mechanism of the Appointment). This appears as a common transition (synchronization event) which can be fired only if the nets to be synchronized have the necessary resources (necessary tokens to the firing of the transition). Susanna Donatelli studied this concept of synchronous composition on GSPN, and called it **Superposition of transitions** (Donatelli 1994).

Def 22: Synchronous composition operator (Moreaux 1996) The GSPN N = (P, T, Pre, Post, Inh, Pri, M_0, θ) is a **synchronous composition** of K subnets N_k = (P_k, T_k, Pre_k, $Post_k$, Inh_k, Pri_k, $M_{0,k}$, θ_k), denoted $\blacktriangleright\blacktriangleleft_{Ts}(N_1, N_2,...,N_K)$ iff:

1. $(P_k)_{k\in K}$ is a partition of P.
2. T=TS∪TL, where TS≠θ is the set of synchronization timed transitions, and TL is the set of local transitions.
3. $T_k = {}^\bullet P_k \cap P_k{}^\bullet$ and $(T_k \cap TL)_{k\in K}$ is a partition of TL.
4. The functions Pre_k, $Post_k$, Inh_k, Pri_k,θ_k are the restrictions to P_k and T_k of corresponding functions of N.
5. $M_0 = (M_{0,k})_{k\in K}$ (initial marking).

In each GSPN N_k, $T_k=TL_k\cup TS_k$ where TS_k is the set of ***synchronization transitions*** of N_k, and $TL_k=T_k\setminus TS_k$ is the set of local transitions of N_k. A marking M of S is a tuple (M_1,….M_k) of markings of S_k. We denote $r_k = |TRS_k|$.

Example 8 : $T_S=\{t_{12}\}$.



[Fig. 8 : Synchronous Composition of GSPN]

4.2.4. Asynchronous composition operator ▼

The asynchronous composition of GSPNs models communicating subsystems via entities (tasks, resources …). These subsystems communicate via common places (or merged places), by exchange of tokens which are forwarded from a net to another by firing ***output transitions***. The output transitions are those which have these places as post conditions.

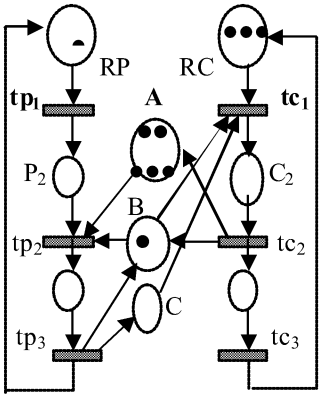
The asynchronous composition of the GSPN and other Petri Nets was studied in several papers (Buchholz 1992), (Buchholz 1993), (Buchholz 1997).

Def 23: Asynchronous composition operator (Moreaux 1996) N=(P,T,Pre,Post,Inh,Pri,M_0, θ) is the **asynchronous composition of** K subnets, denoted $\blacktriangledown_{Ta}(N_1, N_2, \ldots, N_K)$, where N_k=(P_k,T_k,Pre_k,$Post_k$,Inh_k,Pri_k, $M_{0,k}$,θ_k) iff :

1. $T = ?_{k\in K}\ T_k$ (partition of transitions) ;
2. $P = ?_{k\in K}\ P_k$ (partition of places) ;
3. $\forall k\in K,\ \forall t\in T_k\ \ {}^\bullet t \cup t^\bullet \subseteq P_k$

A marking M of N is $M = M(P_k)=(M_k)_{k\in K}$ and $TO_k=\{t\in T_k \mid t^\bullet \cap (P\setminus P_k)\neq \phi\}$ is the set of output transitions of N_k.

Example 9 :



[Fig.9: Asynchronous Composition of GSPN : Producers and consumers]

4.2.5. Hide Operator or encapsulation operator

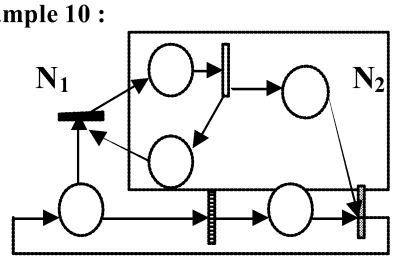
It is sometimes interesting to hide certain actions, and to make abstraction of them. In this objective, we introduce the **encapsulation operator** which **hides** a Petri subnet in another.

Def 24 : Hide operator

N_1 **encapsulate** or **hide** N_2 if and only if :

- $P_2 \subset P_1$, $T_2 \subset T_1$
- $Pre_2(p,t) = pre_1(p,t)$ if $p\in P_2$ and $t\in T_2$
- $Post_2(p,t) = post_1(p,t)$ if $p\in P_2$ and $t\in T_2$
- $Inh_2(p,t) = Inh_1(p,t)$ if $p\in P_2$ and $t\in T_2$
- $Pri_2(t) = pri_1(t)$ if $t\in T_2$, and $\theta_2(t) = \theta_1(t)$ if $t\in T_2$

Example 10 :



[Fig. 10 : Hide operator]

4.2.6. Union Operator

The union between two Petri subnets represents the concatenation of the two subnets.

Definition 25: Union operator

$N_1\cup N_2$ is a GSPN $N=(P_1\cup P_2, T_1\cup T_2, Pre, Post, Inh, Pri, M_0, \theta)$

- Pre : $(P_1\cup P_2,\ T_1\cup T_2)\rightarrow |N$ so that :
  $Pre(p,t) = pre_1(p,t)$ if $p\in P_1$ and $t\in T_1$
  $Pre(p,t) = pre_2(p,t)$ if $p\in P_2$ and $t\in T_2$, and 0 otherwise.
- Post : $(P_1\cup P_2,\ T_1\cup T_2)\rightarrow |N$ so that:
  $Post(p,t) = post_1(p,t)$ if $p\in P_1$ and $t\in T_1$
  $Post(p,t) = post_2(p,t)$ if $p\in P_2$ and $t\in T_2$, and 0 otherwise.
- Inh : $(P_1\cup P_2,\ T_1\cup T_2)\rightarrow |N$ so that:
  $Inh(p,t) = Inh_1(p,t)$ if $p\in P_1$ and $t\in T_1$
  $Inh(p,t) = Inh_2(p,t)$ if $p\in P_2$ and $t\in T_2$, and 0 otherwise.
- Pri : $(T_1\cup T_2)\rightarrow |N$ so that: $Pri(t) = pri_1(t)$ if $t\in T_1$,
  $Pri(t) = pri_2(t)$ if $t\in T_2$, $Pri(t) = 0$ otherwise.
- θ : $(T_1\cup T_2)\rightarrow |N$ so that: $\theta(t) = \theta_1(t)$ if $t\in T_1$,
  $\theta(t) = \theta_2(t)$ if $t\in T_2$, $\theta(t) = 0$ otherwise.

Once our algebra defined, we present next our approach of decomposition.

5. DECOMPOSITION APPROACH

The decomposition approach is performed in two steps:
1. We partition the SPN/GSPN analysed into several layers, each layer represents the possible actions which can be executed at the n^{th} step of system life.
2. On the basis of multilayer view, we search special subnets and build progressively an algebraic expression linking identified subnets with operators defined above.

5.1. Multilayer Decomposition Of a GSPN

The principle is to structure the Petri net modelling a system X, into several **layers** or **levels :** a layer i represents the i^{th} step of system life and is given by possible actions at this step, supplied with their necessary resources.
In terms of Petri Nets, a layer i characterizes the possible firing of one or more transitions enabled simultaneously. Thus, it is constituted of places marked at an instant t_i, and their output transitions (immediate successors). Once this structuring made, the layers are inspected to possibly find subnets and operators linking them to form the entire SPN.
We note that the proposed approach for structuring a Petri net into layers is defined as well for ordinary Petri nets, SPN or GSPN. In the following, we introduce this approach for an ordinary Petri Net (PN).

- **Structuring a PN into layers**

Definition 26: Level of a node
We call **level** of a node x, denoted **v(x)**, the maximum length an elementary path whose end is x.
Definition 27: Enabling Set and Enabling preconditions
Let N be Petri Net : N = (P, T, Pre, Post, Inh, M_0). We call the **enabling set of** T for M_0, **Tenab(T,M_0)**, the set :

$\{t \in T / \forall p \in P$ where $Pre(p,t) \neq 0, M(p) >= Pre(p,t)\}$

For an enabling set T'=Tenab(T, M_0), we associate the **Enabling preconditions**, **Penab(T',M_0)**, defined as :

$\{p \in P / \forall t \in T$ where $Pre(p, t) \neq 0, M(p) >= Pre(p,t)\}$.

Proposition 1 : Let G=(X, E) be a graph without circuits. So, the set of nodes X is a partition $\{C_0 \cup C_1 \cup \ldots \cup C_p\}$ so that : $x \in C_i \Leftrightarrow v(x)=i$:
C_0 ={ Sources of G}
C_1 ={ Sources of G-C_0 }, ...
C_p ={ Sources of G- $\cup_{j=0}^{i-1} C_j$ }
Basing on this proposition, we can state the following :
Proposition 2 : Let N be Petri Net : N =(P, T, Pre, Post, Inh, M_0) whose graph G=(X, E). So, N admits a structure of layers $\{C_0 \cup C_1 \cup \ldots \cup C_p\}$ so that :
$x \in C_i \Leftrightarrow (x=p, p \in P / v(p)=2i)$ or $(x=t, t \in T / v(t)=2i+1)$.
Furthermore, if G is with or without circuits, we can have : $\exists$ i, j, $C_i \cap C_j \neq \emptyset$ because off the constitution of layers.

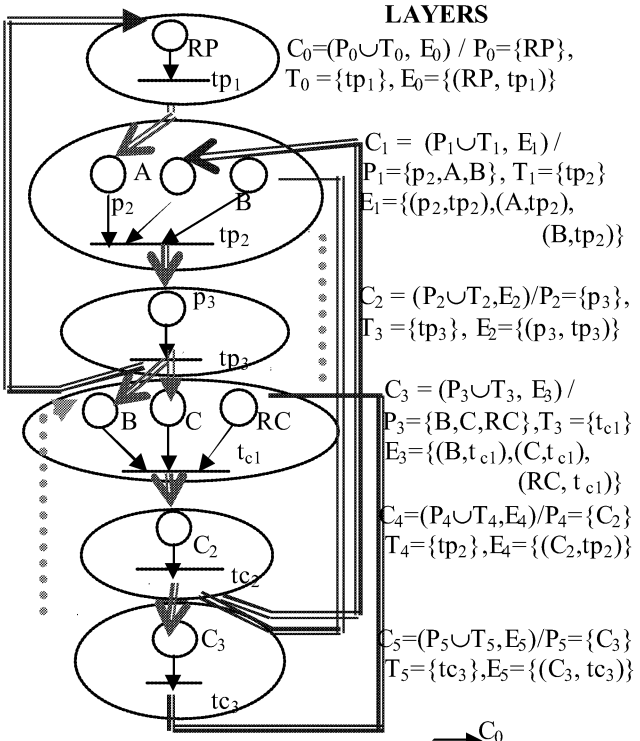
N can be decomposed into layers by the following manner:
Layers Construction Algorithm
Begin
1. C_0 =(X_0, E_0) / X_0= $P_0 \cup T_0$ where T_0=Tenab(T, M_0) and P_0=Penab(T_0,M_0), and E_0=$\{e \in E / \exists x \in X_0, \exists y \in X_0, e=(x, y)\}$.
2. E= E-E_0 ,
3. i=1
4. Repeat
   - Let M_i be the marking obtained from M_{i-1} when all transitions of T_{i-1} are fired..
   -C_i=(X_i,E_i)/ X_i=$P_i \cup T_i$,T_i=Tenab(T,M_i), P_i=Penab(T_i,M_i), E_i=$\{e \in E / \exists x \in X_i, \exists y \in X_i, e=(x, y)\}$.
   - Let L_i be the set of arcs linking C_{i-1} to C_i. We call L_i the **Link** between C_0 and C_1, denoted L(C_0,C_1) :
   L_i=L(C_{i-1}, C_i)=$\{e \in E / \exists x \in X_{i-1}, \exists y \in X_i, e=(x, y)\}$.
   - E= E-E_i , E= E-L_i, i=i+1.

until E=$\emptyset$
End

- **Link tree of a structure of layers of a Petri net**

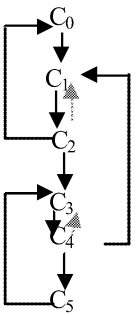
Let p be the number of layers obtained for a PN N.
We call **Link Tree** of a PN N the graph A=(C, L) where :
- C= { C_0, C_1, …, C_p}
- L= { L_1, …, L_p}$\cup$ L' / L' is the set of arcs linking the none adjacent layers.

Example 11 : Let us apply the algorithm above on Fig.9.



- **Corresponding Link Tree**

L_1 = L(C_0, C_1)={(tp_1, p_2)}
L_2 = L(C_1, C_2)={(tp_3, B)}
L_3 = L(C_2, C_3)={(tp_1, p_2)}
L_4 = L(C_3, C_4)={(tc_1, C_2)}
L'$_4$= L(C_4, C_2)={ (tc_2, A), (tc_2, B)}
L_5 = L(C_4, C_5)={(tc_2, C_3)},
L'$_5$= L(C_5, C_3)={(tc_3, RC)}



5.2.How to find the algebraic expression of a Petri Net

Let G the graph of a Petri Net N = (P,T,Pre,Post,Inh,M_0) / G=(X,E). Let np be the number of layers obtained for N, and A=(C,L) the corresponding link tree: C= {C_0,C_1,..,C_{np}} / C_i=(X_i,E_i), X_i=$P_i \cup T_i$ and, L= { L_1, …, L_{np}}$\cup$ L'
In order to find the algebraic expression corresponding to a Petri Net, we introduce the following concept :
Definition 28 : Derived Net Before / Net After of a node
Let N be a marked Petri Net N = (P, T, Pre, Post, Inh, M_0) to which corresponds the graph G=(X,E).
W call **Derived Net after (NetAfter(x))** of a node $x \in X$ (resp. **Derived Net before** of x, **NetBefore(x)**), the set of successors (resp. predecessors) of x and the successors (resp. predecessors) of each element of the set. These nets are built with a recursive manner :

1 . Building NetAfter(x)= (X_a, E_a) /
Computation of X_a : $X_a=\{y\in x^\bullet\}$
Repeat $S=X_a$, $y\in S$, $X_a= X_a\cup\{z\in y^\bullet\}$, $S=S-\{y\}$ **until** $S=\emptyset$
Computation of E_a : $E_a=\{e\in E/\exists\ x\in X_a, \exists\ y\in X_a, e=(x,y)\}$
2. Building NetBefore(x)= (X_b, E_b) /
Computation of X_b : $X_b=\{y\in {}^\bullet x\}$
Repeat $S=X_b$, $y\in S$, $X_b= X_b\cup\{z\in {}^\bullet y\}$, $S=S-\{y\}$ **until** $S=\emptyset$
Computation of E_b : $E_b=\{e\in E/\exists\ x\in X_b, \exists\ y\in X_b, e=(x,y)\}$

Now, we can identify subnets and related algebraic expression of a Petri net with this algorithm :

Decomposition Algorithm

Begin

1. Identify in each layer subnets completely independent, ie: for a layer C_l, find subnets $SN_{jl}=(P_{jl}, T_{jl}, Pre_{jl}, Post_{jl}, Inh_{jl}, Pri_{jl}, M_{0jl}, \theta_{jl})$ so that : $\cap_{j1=1}^{k1} P_{jl}=\emptyset$ and $\cap_{j1=1}^{k1} T_{jl}=\emptyset$.
We can deduce a first algebraic expression binding the subnets of the layers as follows:
Let $C_0=\{SN_{j0} / j_0=1,..., k_0\}$,
$C_1=\{SN_{j1} / j_1=1,..., k_1\}$, …,
$C_{np}=\{SN_{jnp} / j_{np}=1,..., knp\}$. So, expression **(1)** is :
$N=(||_{j0=1}^{k0} SN_{j0}) \blacktriangledown_{Ta0} (||_{j1=1}^{k1} SN_{j1}) \blacktriangledown_{Ta1} ... \blacktriangledown_{Tanp} (||_{jnp=1}^{knp} SN_{jnp})$
where T_{ai}, i=1 à np : set of transitions linking between the subnets $SN_{j(i-1)}$ and SN_{ji} by an asynchronous composition.

2.a/ Find places which are duplicated in more than one layer: As the construction of layers is based on activity (i.e. transition) to describe a step of system execution, we can have the case of a place participating with several activities or transitions of different layers. To palliate to that, it is necessary to detect these duplicated places, and gather their to make only one subnet.
b/Recompute links L between new layers and the link tree.
c/ For each link l_i between two layers C_k et C_j linking two subnets SN_{ku} and SN_{ju} by a transition t : $SN_{ku} \blacktriangledown_{\{t\}} SN_{ju}$

3. Check in each C_l if **subnets of type NCSN** exist, by exploring for each subnet their successors in next layers.

4. Check the existence of **subnets of type SN_nPT,** i.e. find transitions t having $d^-(t)>1$, let $d^-(t)=k$.
- Compute $t^\bullet$ and ${}^\bullet t$.
- For each place p_u de ${}^\bullet t$, compute :
NetBefore(p_u)= (X_{SNu}, E_{SNu})/ $X_{SNu}= P_{SNu}\cup T_{SNu}$.
- Compute the subnets NetAfter(t)= (X_{SNv}, E_{SNv}) / $X_{SNv}=P_{SN}\cup T_{SNv}$ corresponding to each arc $e_v=(t, p_v)$.

If $d^+(t)=1$
Then If $\cap_u P_{SNu}=\emptyset \wedge \cap_u T_{SNu}=\emptyset$
Then ? $_{\{t\}}$[[? ? $_{\{t\}}$ (NetBefore(p_{u1}), NetBefore(p_{u2}), …, NetBefore(p_{uk}))], NetAfter(t)] **....(2)**
Else - Group subnets NetBefore(p_u) whose intersection≠∅.
- Replace in (2) these subnets by their **union**.
- If ∩ of these subnets≠∅, the expression does not contain **Synch operator**, but **Asynch (?)** operator linking the union of subnets NetBefore and NetAfter :
? $_{\{t\}}$**(** (NetBefore(p_{u1}) ∪ NetBefore(p_{u2}) ∪ … ∪ NetBefore(p_{uk2})) , NetAfter(t)**).**
Endif
Else see case 5°/ **SN_nPTmP.**
Endif

Once the expression deduced ((2) or (2) modified), to pursue the application of the algorithm, it is necessary to abstract subnets included in the new expression, in order to find a global expression connecting these subnets to the remainder of the initial model. So, we replace these subnets by a macro subnet ABS, and compute again the new layers and corresponding links, considering the subnet ABS and not its content. Then, we deduce relations between subnets of new layers, by using the **Asynch**(?) operator. This step is called **Abstraction step**.

5. Check if subnets of **type SN_nPTmP** exist, i.e. find the transitions t having $d^+(t)=k_1>1$, and $d^-(t)=k_2>1$.
- Compute $t^\bullet$ and ${}^\bullet t$.
- For each place p_u de $t^\bullet$, compute :
NetBefore(p_u)= (X_{SNu}, E_{SNu}) / $X_{SNu}= P_{SNu}\cup T_{SNu}$.
- Compute the subnets $NetAfter_v(t)$= (X_{SNv}, E_{SNv}) / $X_{SNv}=P_{SNv}\cup T_{SNv}$ corresponding to each arc $e_v=(t, p_v)$.

If $(\cap_u P_{SNu}=\emptyset \wedge \cap_u T_{SNu}=\emptyset) \wedge (\cap_v P_{SNv}=\emptyset \wedge \cap_v T_{SNv}=\emptyset)$
Then Distribute the set of subnets NetAfter between the subnets NetBefore so that the union is made between a subnet NetAfter and another NetBefore.
Let k_1 subnets $NetAfter_v(t)$, and k_2 subnets NetBefore(p_u).
If $k_1=k_2$
Then ? ? $_{\{t\}}$(NetBefore(p_{u1})∪$NetAfter_{v1}$(t),NetBefore(p_{u2}) ∪$NetAfter_{v2}$(t),…,NetBefore(p_{uk1}) ∪$NetAfter_{k1}$(t))…**(3)**
Else If $k_1<k_2$
Then We have the expression:
? ? $_{\{t\}}$ (NetBefore(p_{u1}) ∪ $NetAfter_{v1}$(t), …,NetBefore(p_{uk1}) ∪ $NetAfter_{k1}$(t), …, NetBefore(p_{uk2}) ∪ $NetAfter_{k2}$(t)) ....**(4)**
Else /* $k_1>k_2$ */ We have the expression:
? ? $_{\{t\}}$ (NetBefore(p_{u1}) ∪ $NetAfter_{v1}$(t), …,NetBefore(p_{uk2}) ∪ **(**$NetAfter_{k2}$(t) || …|| $NetAfter_{k1}$(t)**)**) …. **(5)**
Endif
Endif
ElseIf $\cap_u P_{SNu}=\emptyset \wedge \cap_u T_{SNu}=\emptyset$ /*$\cap_v P_{SNv}\neq\emptyset \vee \cap_v T_{SNv}\neq\emptyset$*
Then - Group the subnets NetAfter whose intersection≠∅.
- Replace in (3), (4) or (5) these subnets by **their union.**
ElseIf $\cap_v P_{SNv}=\emptyset \wedge \cap_v T_{SNv}=\emptyset$ /* $\cap_u P_{SNu}\neq\emptyset \wedge \cap_u T_{SNu}\neq\emptyset$*
Then Group subnets NetBefore(p_uc
- Replace in (2) these subnets by their **union**.
- If ∩ of these subnets≠∅, the expression will not contain the Synch (**? ?)** operator, but rather the **? operator :**
? $_{\{t\}}$((NetBefore(p_{u1}) ∪ NetBefore(p_{u2}) ∪ … ∪ NetBefore(p_{uk1}), (NetAfter$_1$(t) || …|| $NetAfter_{k1}$(t)).
Else/*$(\cap_u P_{SNu}\neq\emptyset \vee \cap_u T_{SNu}\neq\emptyset) \wedge (\cap_v P_{SNv}\neq\emptyset \vee \cap_v T_{SNv}\neq\emptyset)$
? $_{\{t\}}$**(** (NetBefore(p_{u1}) ∪NetBefore(p_{u2}) ∪…∪ NetBefore(p_{uk2})) , (NetAfter$_1$(t) ∪ …∪ $NetAfter_{k1}$(t))**).**
Endif
Endif
Endif

Once the expression deduced, do the abstraction step.

6. Check if subnets of **type SN_PnT** exist, having places p / $d^+(p)=k>1$: if p ∋, check if ∀ $t_u\in p^\bullet$, $^\bullet t_u=\{p\}$ exactly.
- Compute $p^\bullet$.
- For each transition t_u of $p^\bullet$, compute :
NetAfter(t_u)= (X_{SNu}, E_{SNu})/ $X_{SNu}= P_{SNu}\cup T_{SNu}$.

If $\cap_u P_{SNu}=\emptyset \wedge \cap_u T_{SNu}=\emptyset$
Then We denote SR the subnet SN_PnT. We have
? $_{\{tu1, tu2, ..., tuk\}}$(SR, NetAfter($t_{u1}$)+ NetAfter($t_{u2}$)+…+ NetAfter($t_{uk}$)) …. **(6)**
Else - Group subnets NetAfter(t_u) whose intersection≠∅.
- Replace in 6) these subnets by their **union**.
Endif

Once the expression deduced, do the abstraction step

7. Check if subnets of **type SN_nTP** exist, i.e. find the places p having $d^-(p)>1$. Let $d^-(p)=k$.

- Compute $p^{\bullet}$.
- Compute the subnets $NetBefore_u(p) = (X_{SNu}, E_{SNu})$ / $X_{SNu}=P_{SNu} \cup T_{SNu}$, each one has $Intf^{+}(NetBefore_u(p)) = \{t_u\}$ constituted of one transition t_u de p .

If $\cap_u P_{SNu} = \varnothing \wedge \cap_u T_{SNu} = \varnothing$

Then We denote SR the subnet whose $Intf(SR)=\{p\}$. So:

? $_{\{tu1, tu2, ..., tuk\}}$ ($NetBefore_{u1}(p) \parallel NetBefore_{u2}(p) \parallel ... \parallel NetBefore_{uk}(p)$, SR) .... **(7)**

Else Group subnets $NetBefore_u(p)$ whose intersection$\neq\varnothing$.

- Replace in (7) these subnets by their **union**.
- If $\cap$ of these subnets$\neq\varnothing$, then the operator $\parallel$ is not used.

Endif

Once the expression deduced, do the abstraction step.

8. Check the existence of subnets of **type SN_TnP,** i.e. find the transitions t having $d^{+}(t)>1$. Let $d^{+}(t)=k$.

- Compute $t^{\bullet}$.
- Compute $NetAfter_u(t) = (X_{SNu}, E_{SNu})$/ $X_{SNu}=P_{SNu} \cup T_{SNu}$, each one has as $Intf(NetAfter_u(t)) = \{p_u\}$.

If $\cap_u P_{SNu} = \varnothing \wedge \cap_u T_{SNu} = \varnothing$

Then We denote SR the subnet whose $Intf^{+}(SR)=\{t\}$. So:

? $_{\{t\}}$ (SR, $NetAfter_{u1}(p) \parallel NetAfter_{u2}(p) \parallel ... \parallel NetAfter_{uk}(p)$) .... **(8)**

Else - Group subnets $NetBefore_u(p)$ whose intersection$\neq\varnothing$.

- Replace in (8) these subnets by their **union**.
- If $\cap$ of these subnets$\neq\varnothing$, then the operator $\parallel$ is not used.

Endif

Once the expression deduced, do the abstraction step.

9. Check if subnets of **type SN_nTPmT** exist, i.e. find the places p having $d^{+}(p)=k_1>1$ and $d^{-}(p)=k_2>1$.

This is the same as case 6, deduce associated expression which gives a macro subnet, this latter abstracts subnets included in the expression, and finally study case 7.

10. If all subnet types have been checked, we write the final expression of N by using expressions found with relations deduced from the last computed link tree.

End

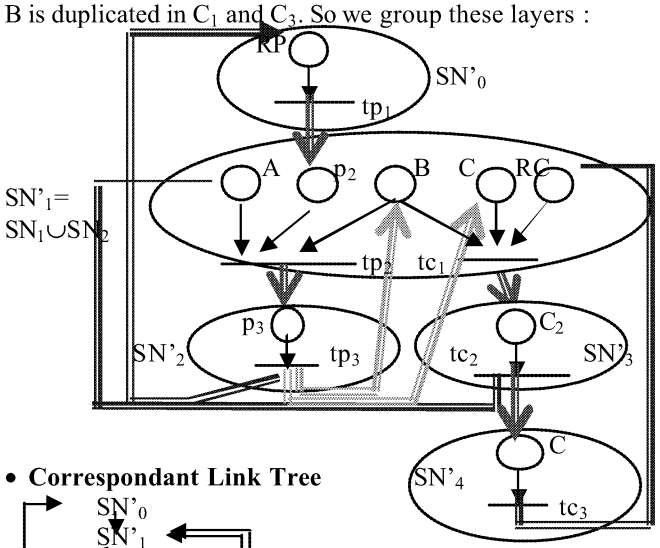
Previous example : Producers and consumers

1. 1st expression

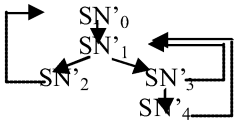
$N=SN_0 \blacktriangledown_{\{tp1\}} SN_1 \blacktriangledown_{\{tp2\}} SN_2 \blacktriangledown_{\{tp3\}} SN_3 \blacktriangledown_{\{tc1\}} SN_4 \blacktriangledown_{\{tc2\}} SN_5$.

2. Find places duplicated and group their subnets :

B is duplicated in C_1 and C_3. So we group these layers :



- **Correspondant Link Tree**



- **Relations deduced between the subnets :**

$SN'_0 \blacktriangledown_{\{tp1\}} SN'_1$, $SN'_1 \blacktriangledown_{\{tp2\}} SN'_2$,
$SN'_2 \blacktriangledown_{\{tp3\}} SN'_0$, $SN'_1 \blacktriangledown_{\{tp3\}} SN'_3$,
$SN'_3 \blacktriangledown_{\{tc1\}} SN'_4$, $SN'_3 \blacktriangledown_{\{tc2\}} SN'_1$, $SN'_4 \blacktriangledown_{\{tc3\}} SN'_1$

3. Check in each C_1 if **subnets of type NCSN** exist: None.

4. Check if subnets of **type SN_nPT** exist.

2 subnets **SN_nPT** exist : SN_1, SN_2 which compose SN'_1.

a) For SN_1 : We compute NetBefore $(p_2) = SN'_0$,
NetBefore(A)=SN'_3 , NetBefore(B)=SN'_2, SN'_3,
NetAfter$(tp_2)=SN'_2$. So, we deduce :

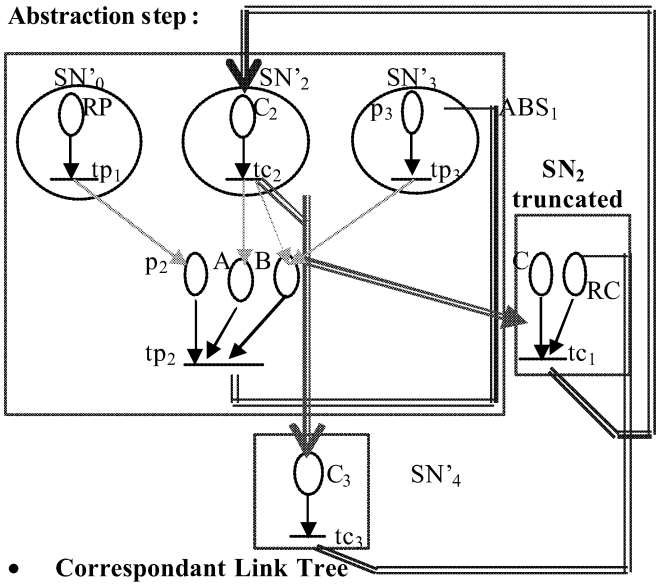
? $_{\{tp2\}}$[? ? $_{\{tp2\}}$**($SN'_0 \cup S_1$, $SN'_3 \cup S_2$, $SN'_2 \cup S_3$), SN'_2]**

$S_1=(P_{s1} \cup T_{s1}, E_{s1})$ / $P_{s1}=\{p_2\}, T_{s1}=\{tp_2\}, E_{s1}=\{(p_2, tp_2)\}$.

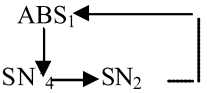
$S_2 = (P_{s2} \cup T_{s2}, E_{s2})$ / $P_{s2}=\{A,B\}$, $T_{s2}=\{tp_2\}$, $E_{s2}=\{(A,tp_2), (B,tp_2)\}$.

$S_3= (P_{s3} \cup T_{s3}, E_{s3})$ / $P_{s3}=\{B\}$, $T_{s3}=\{tp_2\}$, $E_{s3}=\{(B, tp_2)\}$.

Abstraction step :



- **Correspondant Link Tree**



- **Relations deduced between the subnets :**

$ABS_1 \blacktriangledown_{\{tc2\}} SN'_4$, $SN'_4 \blacktriangledown_{\{tc3\}} SN_2$, $SN_2 \blacktriangledown_{\{tc1\}} ABS_1$

b) For SN_2 : We compute NetBefore (B) = ABS_1
NetBefore(C)= ABS_1 , NetBefore(RC) = SN'_4
NetAfter(tc_1) = ABS_1 . So, we deduce :

? $_{\{tc1\}}$ [? ? $_{\{tc1\}}$ **($ABS_1 \cup S_4$, $SN'_4 \cup S_5$), ABS_1]**

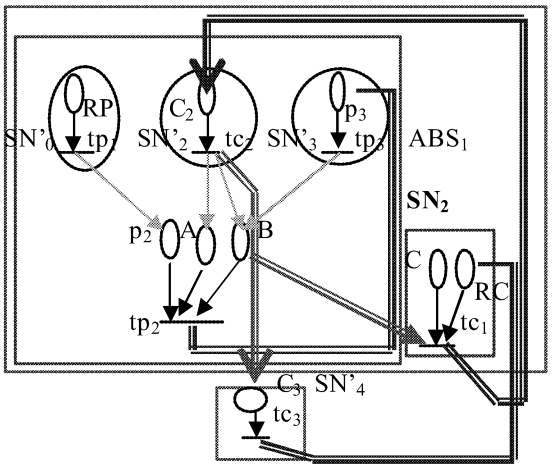
Where : $S_4=(P_{s4} \cup T_{s4}, E_{s4})/P_{s4}=\{C\}, T_{s4}=\{tc_1\}, E_{s4}=\{(C,tc_1)\}$

$S_5= (P_{s5} \cup T_{s5}, E_{s4})$ / $P_{s4}=\{RC\}$, $T_{s5}=\{tc_1\}$, $E_{s5}=\{(RC,tc_1)\}$.

Global Expression :

? $_{\{tc1\}}$ [? ? $_{\{tc1\}}$ [[? $_{\{tp2\}}$ [? ? $_{\{tp2\}}$ **($SN'_0 \cup S_1$, $SN'_3 \cup S_2$, $SN'_2 \cup S_3$), SN'_2]** $\cup S_4$, $SN'_4 \cup S_5$] , [? $_{\{tp2\}}$ [? ? $_{\{tp2\}}$ **($SN'_0 \cup S_1$, $SN'_3 \cup S_2$, $SN'_2 \cup S_3$), SN'_2]**]

Abstraction step : ABS_2



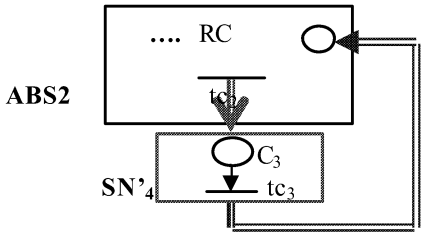
- **Correspondant Link Tree**



- **Relations deduced between the subnets :**

$ABS_2 \blacktriangledown_{\{tc2\}} SN'_4$, $SN'_4 \blacktriangledown_{\{tc3\}} ABS_2$

5. Check the existence of subnets of **type SN_nPTmP,** i.e. find the transitions t having $d^+(t)>1$: In the new abstracted subnet, there are not subnets of type **SN_nPTmP.**



6. Check the existence of subnets of **type SN_PnT,** i.e. find the places p having $d^+(p)>1$: In the new abstracted subnet, there are not subnets of type **SN_PnT.**

7. Check the existence of subnets of **type SN_nTP,** i.e. find the places p having $d^+(p)>1$: In the new abstracted subnet, there are not subnets of type **SN_nTP.**

8. Check the existence of subnets of **type SN_TnP,** i.e. find the transitions t having $d^+(t)>1$: In the new abstracted subnet, there are not subnets of **SN_TnP.**

9. End of application of the algorithm.

Final Expression of the SPN N

By using the expression found and the deduced relations from the last link tree, we have two global expressions of N, that determine two possible behaviors of the PN

1st expression : $? \ {}_{\{tc2\}} \{ ? \ {}_{\{tc1\}} [? \ ? \ {}_{\{tc1\}} [[? \ {}_{\{tp2\}} [? \ ? \ {}_{\{tp2\}} (SN'_0 \cup S_1 \ , SN'_3 \cup S_2, SN'_2 \cup S_3), SN'_2] \cup S_4 , SN'_4 \cup S_5] , [? \ {}_{\{tp2\}} [? \ ? \ {}_{\{tp2\}} (SN'_0 \cup S_1, SN'_3 \cup S_2, SN'_2 \cup S_3), SN'_2]], SN'_4 \}$

2nd expression : $? \ {}_{\{tc3\}} \{ SN'_4 , ? \ {}_{\{tc1\}} [? \ ? \ {}_{\{tc1\}} [[? \ {}_{\{tp2\}} [? \ ? \ {}_{\{tp2\}} (SN'_0 \cup S_1 \ , SN'_3 \cup S_2, SN'_2 \cup S_3), SN'_2] \cup S_4 , SN'_4 \cup S_5] , [? \ {}_{\{tp2\}} [? \ ? \ {}_{\{tp2\}} (SN'_0 \cup S_1, SN'_3 \cup S_2, SN'_2 \cup S_3), SN'_2]] \}$

6. Conclusion

In this paper, a new decomposition technique of SPN and GSPN has been proposed. The aim is to reduce the impact of states space explosion which can appears during performance analysis of a system. Thanks to this method, we can represent the behaviour of the system by an algebraic expression, which links system components (subnets). This expression can be used to compute the global solution of SPN under study as a function of components solutions, avoiding thus analysis of whole system.

As our objective is to simplify performance analysis, the computation of the global solution thanks to the algebraic expression and its operators will be studied as a future work. Qualitative properties of the whole system can also be studied to deduce properties from subnets properties and their algebraic expression.

7. References

Buchholz, P. 1992. "A hierarchical view of GCSPNs and its impact on qualitative and quantitative analysis". *Journal of Parallel and Distributed Computing*, 15(2). 207-224.

Buchholz, P. 1993. "Aggregation and reduction techniques for hierarchical GCSPN". In *Proceedings Of the 5th International Workshop on Petri Nets and Performance Models*. (Toulouse, France, Oct.19-22). IEEE Computer Society Press, 216-225.

Buchholz, P. 1997. "Hierarchical structuring of superposed GSPNs". IEEE, 81-90.

Davio, M. 1981. "Kronecker products and shuffle algebra ". 'Feb.1981). IEEE Trans. on Computers. C-30(2).

Donatelli, S. and Sereno, M. 1991. "On the product form solution for Stochastic Petri Nets". IEEE, 154-172.

Donatelli, S. 1994. "Superposed generalized Stochastic Petri nets : definition and efficient solution". In Proceedings Of the 15th International Conference on Application and Theory of Petri Nets. (Zaragoza, Spain, Jun.20-24). Springer-Verlag, Robert Valette Editor, N°815 in LNCS, 258-277.

Dutheillet, C. and Haddad, S. 1989. "Aggregation of states in colored Petri nets : Application to a multiprocessor architecture". (PARIS, 1989). IEEE, 40-49.

Dutheillet, C. 1992. "Symétries dans les réseaux de Petri colores: Définitions, analyse et application à l'évaluation des performances". Doctorat thesis. (Paris 6 University, March 1992).

Florin, G. and Natkin, S. 1989. " Matrix product form solution for closed synchronised queuing networks". IEEE, 29-37.

Ioualalen, M. Boukala, M.C. and Aissani, A. 1998. "Réductions dans les Réseaux de Petri Stochastiques. Construction du graphe symbolique. Application à l'évaluation des performances". 4ème Colloque Africain sur la recherche en Informatique. (Dakar, Sénégal. Oct.12-15).

Moreaux. P. 1996. "Structuration des chaînes de Markov des Réseaux de Petri Stochastiques". Thèse de doctorat. (Université Paris Dauphine, Dec.1996).

Plateau, B. and Fourneau, J.M. 1991. "A methodology for solving Markov models of parallel systems". Journal of Parallel and Distributed Computing, 12. 370-387.

Haddad, S. Ilié, J.M. Taghelit, M. and Zouari, B. 1995. "Symbolic reachability graph and partial symmetries". Technical Report of Masi lab. N° 95.30, P&M.Curie university . (Paris, Mars 95).

Haddad, S. and Moreaux, P. 1997. "Aggregation and decomposition for performance evaluation of asynchronous product of high level Petri Nets". Rapport de recherche N°102, Lamsade. Université de Paris Dauphine. Mai 1997.

Truffet, L. 1995. " Méthodes de calcul de bornes stochastiques sur des modèles de systèmes et de réseaux ". Doctorat thesis, Paris 6 University, Nov. 1995.

Zuberek,W.M. 1989. "Performance evaluation using unbounded Timed Petri Nets". (Canada, 1989). IEEE. 180-186.

Evaluation of maintenance process of electrical power plants, using Petri Nets

Vrassidas Leopoulos
Charalampos Tsogkas
Evaggelos Champilomatis
Konstantinos Kirytopoulos
National Technical University of Athens
Iroon Polytechniou 9
15780 Zografou

KEYWORDS
Maintenance, Electrical Power Plants, Petri Nets, Simulation.

ABSTRACT

Objective of the present study is the development of a model for the evaluation of maintenance activities of an electrical plant. Maintenance for the electrical plant under consideration is based on a well-defined process for both corrective and planned interventions, thus failure rates for the components of equipment are based on historical data. The model calculates, through simulation, the duration of maintenance work orders, the availability of the equipment and the workload for each group of the maintenance team. For the model development Generalised Stochastic Petri Nets (GSPN) have been used and the software package ARTIFEX.

INTRODUCTION

A major concern of the power-industry is the availability of the plants (Yang, 2004, Gräber, 2004). Breakdowns may result in serious problems for the supply of power to consumers leading in loss of money, bad reputation for the power provider and public disorder (U.S.-Canada Power System Outage Task Force, 2004). Apart from spare capacity the answer to the problem is the proper maintenance (Katchmar, 1997). Efforts to optimise and to restructure the maintenance processes necessitate a work order handling system which includes a planning, calculation and accounting system for work orders, based on work hourly rates, service catalogues and material orders. Evaluation of such systems necessitate simulation techniques (Adamyan and He, 2004) in order to take into consideration the stochastic nature of equipment failures. The duration of maintenance tasks is also uncertain because the necessary activities are known only after the stoppage and inspection of the components (Ogaji, 2002, Tatsiopoulos et al, 1999). Several studies propose models based on Petri Nets for validation of maintenance processes (Schneeweiss, 2001, Yang and Liu, 1998). The resulting models for real maintenance processes are often too complicate to allow comprehensive graphs (Antoniadis and Leopoulos, 2002).

In the present study, the maintenance process is modelled using Generalised Stochastic Petri Nets (GSPN). Because of the locality principle on states and actions (transitions), Petri net models can be locally modified, refined or made coarse, without altering the rest of the model (Proth, 1993). This means that nets can be synthesised using top-down and bottom-up approaches. Top-down analysis is a procedure that, starting with a general model of the maintenance process, leads to the final model through stepwise refinements. In the bottom-up approach modules are produced for every maintenance sub-process and then composed. Both approaches have been used in order to produce a model comprehensive and appropriate for simulation results.

MAINTENANCE PROCESS

Maintenance is defined as a set of activities intended to keep a functional unit in, or to restore it, to a state in which it can perform a required function. It includes activities such as monitoring, tests, measurements, replacements, adjustments, repairs, and in some cases administrative actions (ISO, 1997).

Maintenance of the power plant under consideration is divided into two major types: corrective and planned or scheduled. The former comprises the majority of the maintenance activities and includes two sub-processes, according to the urgency of the job: “Corrective maintenance Priority 1” and “Corrective maintenance Priority 2”. Repairs are made to the equipment only when actual failure occurs aiming to get the equipment into production quickly. The process includes 27 tasks.

The latter is one sub-process “Planned-Scheduled Maintenance”, which includes 29 tasks and is intended to prevent breakdowns and minimise shutdowns through systematic inspection of equipment, adjustments, and scheduled repairs before failures occur. These two types of maintenance activities, and a simplified picture of the included tasks, are depicted in Figure 1.

The order for maintenance work is the core document of the order-processing system, which tracks the order from the time that it is placed until the final accounting. The urgency of the job is illustrated by the priority of the orders, which indicates the desired commencement date of the work. The order process is given in Figure 2 and it is analysed right afterwards:

(1) Making out an order: If maintenance work is required, a written order is issued. The order must include information about (a) the place that the work has to be performed (specification of the site in accordance with the classification system of the power plant), (b) the type of damage occurred, (c) the type of work that has to be performed, (d) the urgency of order processing and (e) the date when maintenance work should be commenced.

(2) Technical clarification: A plant engineer is responsible for the preparation of maintenance measures. Technical clarification includes the following tasks: Determining the scope of maintenance work, information relating to regulations, limiting sizes and materials usage, information relating to documentation.

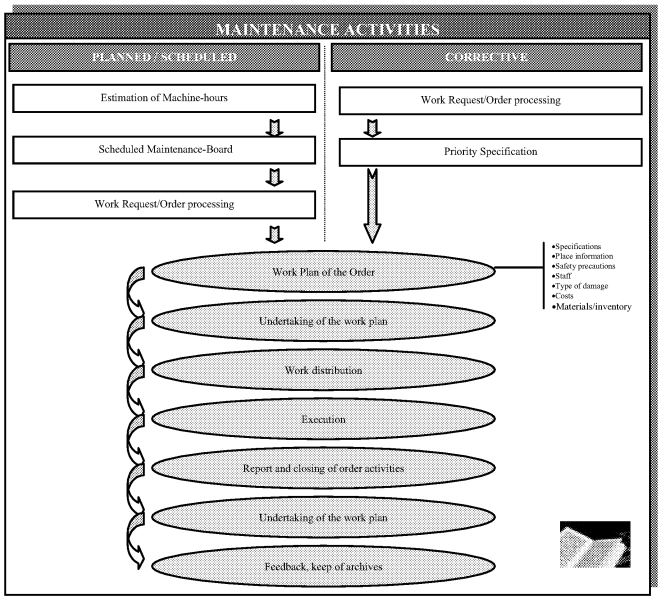


Figure 1: Maintenance activities, types and tasks

(3) Industrial-engineering planning: The maintenance planner prepares Industrial-engineering plans. His task is to prepare a detailed description of the workflow, setting out sequence of operations and the safety measures required. For any maintenance job he has to fulfil the following tasks: Describing the work flow and its work elements, checking whether all required safety measures have been taken determining the target time (job time), ascertaining staff requirements and assigning worker(s), etc.

(4) Controlling and order for delivery: Controlling means to initiate, monitor and take steps ensuring that the maintenance activities will be carried out in the required deadline, quality and costs. The unit in charge of preparing maintenance work passes the order to the unit in charge of executing the work. The following aspects are taken into account: meeting deadlines, optimum utilisation of available capacities and short downtimes.

(5) Execution of the order, acceptance of the service and completion: Before work is commenced, the foreman discusses the order with the worker(s) assigned to do the job. Any deviations from the scope of the work order require prior approval. The plant is ready for operation as soon as the production department has accepted it and all safety precautions have been lifted.

MODELLING, SIMULATION AND RESULTS

A bottom-up (modular) composition method has been used for the development of the first level model. Five subsystems have been modeled and connected namely "Corrective maintenance Priority 1", "Corrective maintenance Priority 2", "Planned-Scheduled Maintenance", "Spare Parts from Warehouse" and "Material Supply". On the first level the five subsystems are presented in Figure 3. The maintenance department has 10 job functions and responsibilities of work. An employee works on a task of a sub-process for the time needed and then he is free to work on a task of another sub-process when necessary.

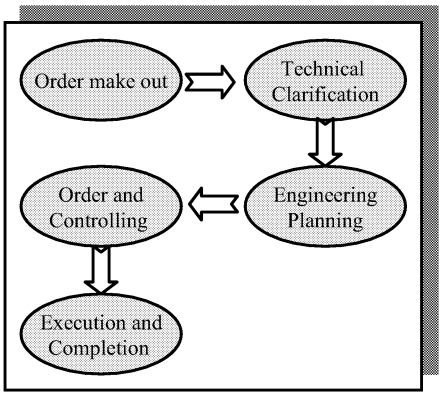


Figure 2: Maintenance order process

Top down analysis begins with an aggregate model of each sub-system and then refinement is done in a stepwise manner to the tasks. The refinement continues until the level of detail satisfies the specification of the system. In Figure 4 a part of the lowest level of refinement is presented for the sub-process "Corrective maintenance Priority 1".

In the results presented in Table 1 the simulation time has been set to 86400 min and the occupation of the maintenance foreman for activities concerning the aforementioned sub-process.

Table 1: Occupation of the foreman for "Corrective maintenance Priority 1"

Run	Occupation of the foreman	% (Sub-process time)	% (Simulation time)
1	4706,37	70,10	5,45
2	4587,62	66,02	5,31
3	4724,72	68,42	5,47
4	4002,16	67,74	4,63
5	4878,02	68,35	5,65
6	4809,11	69,50	5,57
7	4581,09	66,79	5,30
8	4155,20	69,10	4,81
9	4098,82	68,83	4,74

10	4800,58	66,84	5,56
Mean	4534,37	68,14	5,25
SD	325,24		

CONCLUSIONS – FURTHER RESEARCH

This paper presents a case study where Generalised Stochastic Petri Nets (GSPN) have been used for the evaluation of the maintenance activities in power plants. Both top-down analysis and bottom-up synthesis have been used in order to produce a comprehensive model. Simulation is proposed in order to take into consideration the stochastic nature of both failures and duration of maintenance tasks. The model based on the failure rates of the equipment and the duration of maintenance tasks calculates, through simulation, the duration of maintenance orders, the availability of the equipment and the workload for each job function of the maintenance team, enabling the maintenance managers to decide about the workforce size, the resource allocation or the process changes that should take place. Since the modelling of the processes has taken place, a further step should be to model the physical system (components and sub-components of power plants). The interaction between the maintenance processes and the components of the system would thus be possible, leading to results regarding investments in equipment.

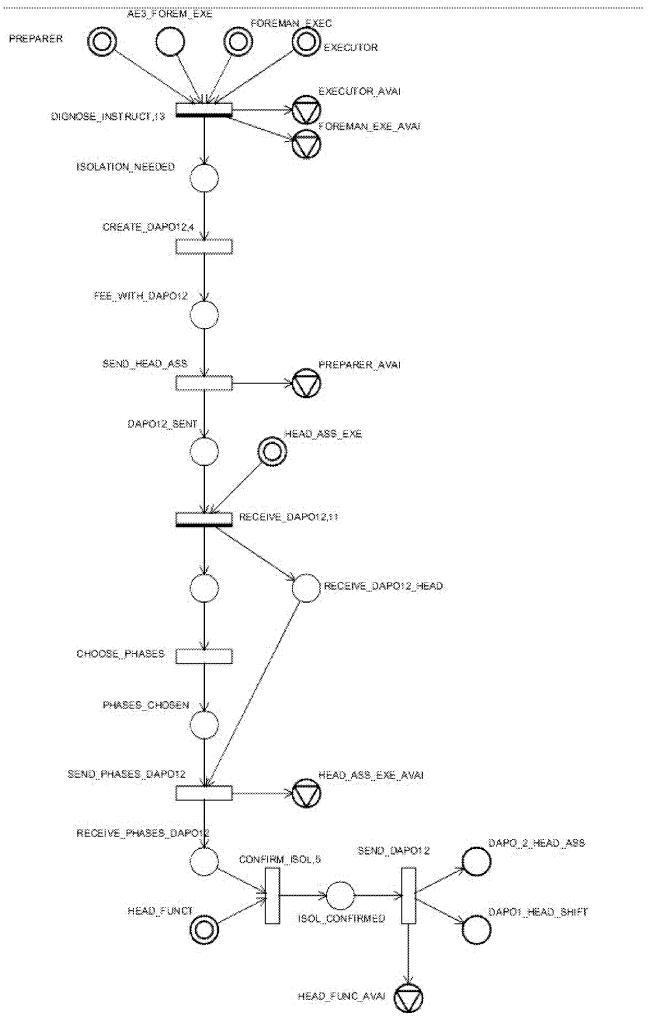


Figure 4: Sub-process "Corrective maintenance Priority 1" procedure 3 – Work request order processing

REFERENCES

Adamyan A. and He D., 2004. "System Failure Analysis Through Counters of Petri Nets Models", Quality and Reliability Engineering International.

Antoniadis I.A. and Leopoulos V.I.N., 2002. "Application of Petri Nets as an operational and reliability analysis tool for criticality based maintenance of industrial equipment, Proceedings of COMADEM.

Gräber U., 2004, "Advanced maintenance strategies for power plant operators—introducing inter-plant life cycle management", International Journal of Pressure Vessels and Piping, In Press.

ISO/ IEC 2382-14, 1997. "Information technology – Vocabulary, Part 14: Reliability, maintainability and availability", ISO.

Katchmar A., 1997. "Reliability Centered Maintenance. Proc. of AISE – Evolving maintenance Strategies", Hamilton.

Ogaji S., Sampath S., Singh R. and Probert D., 2002, "Novel approach for improving power-plant availability using advanced engine diagnostics", Applied Energy, Vol 72 (1), pp. 389-407.

Proth J-M. 1993, "Practice of Petri Nets In Manufacturing", Chapman&Hall.

Schneeweiss W., 2001. "Tutorial: Petri Nets as a Graphical Description Medium for Many Reliability Scenarios", IEEE Transactions on Reliability, vol. 50, no 2.

Tatsiopoulos I.P., Leopoulos V., Panayiotou N.A. and Tsikonis L.K., 1999. "A Comparative Simulation Analysis in the Customer Offer Process using Petri Nets and Discrete Event Simulation", in Graham Horton, Dietmar Moller and Ulrich Rude, 1999, SIMULATION IN INDUSTRY.

U.S.-Canada Power System Outage Task Force, 2004, "Final Report on the August 14th Blackout in the United States and Canada", April 2004, available at https://reports.energy.gov.

Yang S. K., 2004. "A condition-based preventive maintenance arrangement for thermal power plants", Electric Power Systems Research, Vol 72 (1), pp. 49-62.

Yang S.K. and Liu T.S, 1998. "A Petri Net approach to early failure detection and isolation for preventive maintenance", Quality and Reliability Engineering International.

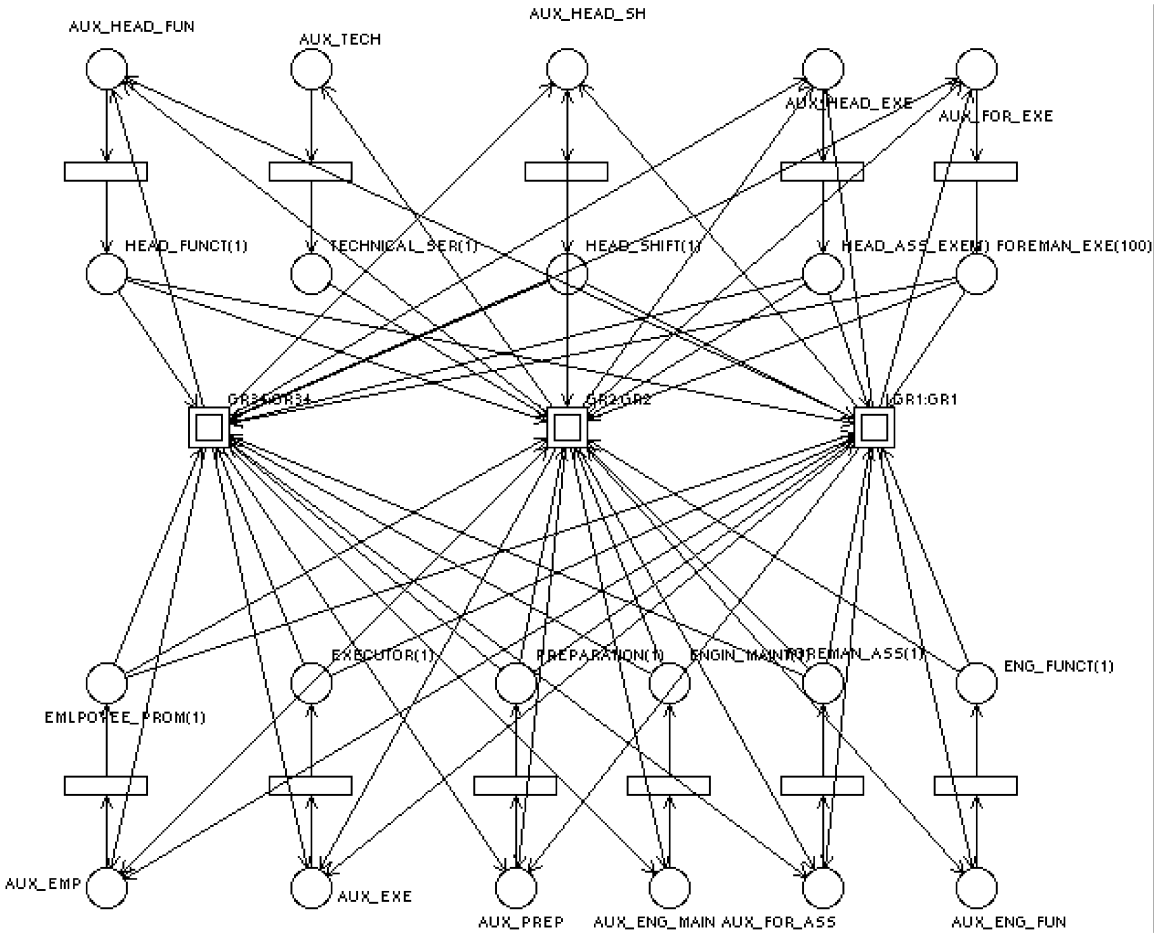


Figure 3: Maintenance processes - first level

TOWARDS THE AUTOMATED CONSTRUCTION OF MULTICOMPONENT PETRI NETS MODELS OF SUPPLY CHAINS

Carmen-Veronica Bobeanu and Hendrik Van Landeghem

Ghent University, Department Industrial Management, Technologiepark 903,
B-9052 Ghent, Belgium
e-mail: carmen.bobeanu@UGent.be
hendrik.vanlandeghem@ UGent.be

KEYWORDS
Petri Nets, Meta-Modelling, Graph Grammars, Supply Chains

ABSTRACT

This paper addresses the authors' current results to generate customised tools supporting the construction of simulation models of complex supply chains. The proposed approach combines the advantages of meta-modelling (to minimise the effort required to construct a tool supporting the implementation of the developed method) and graph transformation systems (to express the rules underlying the handling of the set of routing constructs identified in the authors' work). These provide the basis for further automated generation of complex Petri nets models from the appropriate graph grammars.

1. INTRODUCTION

Nowadays, simulation becomes an increasingly indispensable problem solving methodology for the industrial engineers, operations researchers and management scientists. Although there is a steady progression of simulation users, Banks (Banks 2001) notes that the growth in simulation use is not accelerating. The solution suggested to accelerate the growth is the reduction of the complexity of using the simulation-based tools.
It is the purpose of the research considered in this presentation to provide a natural, simple and powerful method for describing and analysing discrete event systems (DES) not far from the industrial engineers' habitual notions of system design and operation. The proposed approach combines the advantages of meta-modelling (to avoid explicit programming of a customised tool supporting the implementation of the proposed method) (Van Landeghem and Bobeanu 2003) and graph transformation systems (to express models transformation). This contribution revisits the authors' previous work (Bobeanu 2000, Kerckhoffs and Bobeanu 2001, Van Landeghem and Bobeanu 2002) and integrates into a new setting results addressing the systematic bottom-up construction of re-usable Petri nets (PN) models of DES with the use of graph transformations (Ehrig and Padberg 2004) to relate the nets in different development stages.

2. A SYSTEMATIC METHOD SUPPORTING THE BOTTOM-UP CONSTRUCTION OF PN MODELS

In this Section we present and integrate in an overall view the concepts of our systematic approach to construct PN models and the ideas underlying our recent developments in supply chains (SCs) representation and analysis in the PN domain.
We recall that the central idea of our research is a refined PN representation in terms of sets and operations on sets, focusing on the main attributes of a composite PN object specification and anticipating the application of Zeigler's theory (Zeigler et al. 2000) in the PN domain. We recall as well the key elements supporting the manipulation of these abstractions introduced in our previous work (Bobeanu and Alla 1998, Kerckhoffs and Bobeanu 2001):

- *primitive/atomic model*: block component with a well-defined interface, that can be seen as a process with an input transition, T_i, supplying the input of its activity and an output transition, T_o, enabling its evacuation and a place P modelling the temporal entities advance in the net (production place) or constraints to be satisfied (synchronisation place);
- *coupling templates*: standardised means to couple building block components;
- *step-by-step procedure for the construction of compound PN-models*: systematic bottom-up construction of PN models using as input information (about e.g. primitive system components, entity flows, routing constructs, etc.) gathered from the top-down system analysis.

One of the new ideas considered in (Van Landeghem and Bobeanu 2002) is that the development of a well-grounded set of coupling templates forces one to partition the entity flows involved in a SC into three distinct types:

1. orders placed by the customer (denoted as *Ea*);
2. internal orders (denoted as *Eb*);
3. products (denoted as *Ec*).

The formal developments proposed in (Van Landeghem and Bobeanu 2003) as a result of this refinement broaden the applicability of the authors' contribution to also address

the interfacing questions of different flows of entities. In this perspective the following rules are introduced:

I. The **internal coupling templates** of two **sequential models** *A* and *B* will allow a fusion to be applied to a pair of transitions (T_o^a, T_i^b), with T_o^a the output transition of the model *A* and T_i^b the input transition of the model *B*, provided one of the following cases applies:

I.1 The above transitions are enabled for the same type of entities;

I.2 A transformation is allowed between the types of entities enabling the candidate transitions for applying a fusion rule (such as a transition enabled by a component of the information flow and a transition generating a component of the material flow as a result of its firing).

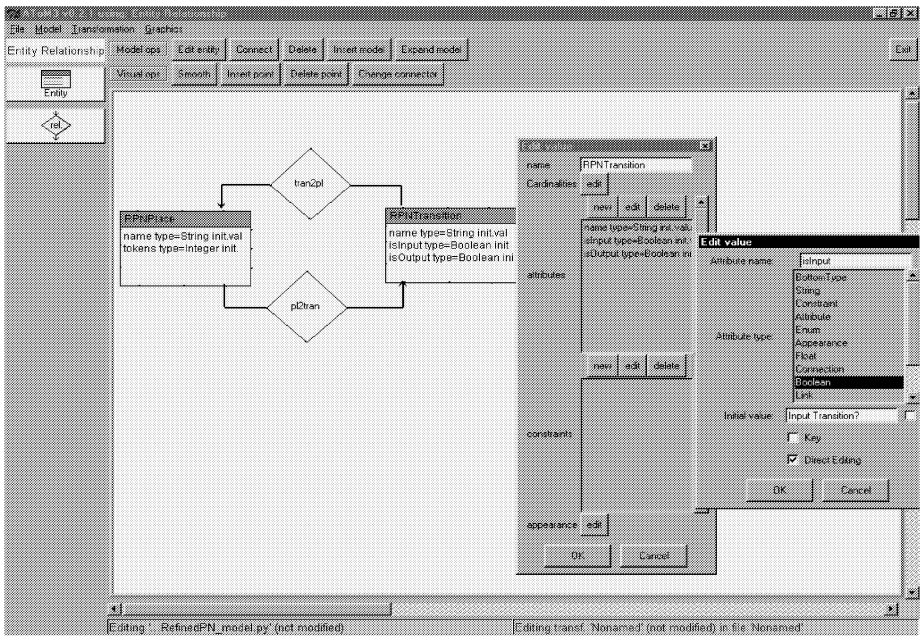
II. A **representative system of the input/ output transitions sets** of a coupled model to be used by an internal coupling while modelling synchronisation and parallelism, will be structured according to the types of entities (or classes identified within the scope of a given type) enabling the addressed input/ output ports. The induced structure will prevent disjoint classes of entities to be supplied/ evacuated using the same input/output ports of the model.

3. A CUSTOM TOOL SUPPORTING STRUCTURAL MODELLING IN THE PN DOMAIN

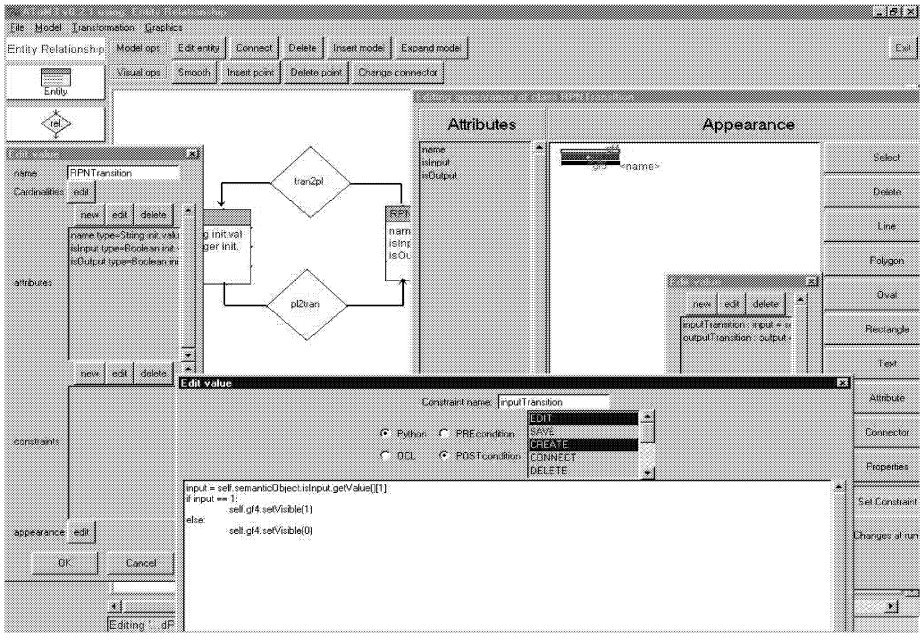
In order to support the above approach, we are developing a modelling tool in the refined formalism proposed in our method. Since the time needed to develop such tools is normally prohibitive, we have chosen for using meta-modelling, as a means to tackle the above problem. *Meta-modelling* is the process of designing modelling languages through meta notations. If we consider a modelling language as a concrete representation of a formalism, a *meta-model* is a model of the formalism itself.

Meta-models specify the *syntax* (the elements of the language) and *semantics* (the meaning) of a class of models. Such a meta-specification allows the automatic generation of a domain-specific modelling and simulation environment to process (create, edit, check, optimise, transform) models described in the specified formalism. The concepts proposed in our developments addressing the modular synthesis of PN-based simulation models of complex DES are implemented in AToM^3, A Tool for Multi-formalism, Meta-Modelling.

The main steps in the generation of the Refined Petri Nets (RPN) formalism from a model designed in the Entity-Relationship formalism, using AToM^3 are explained in (Van Landeghem and Bobeanu 2003). In the sequel we insist on particular steps in the specification of the RPN



(a)



(b)

Figure 1. Setting Up the Properties of a Transition Entity: (a) The Attributes of a Transition Entity; (b) The Appearance of a Transition Entity

formalism supporting the implementation of the extensions to the PN formalism introduced in our work.
The refinement of the transitions set envisaged in our formalism was implemented by adding two Boolean variables (isInput and isOutput) to the transition entity, allowing the user to specify if the current transition is an input or output port, respectively, of the model (see Figure 1 (a)). The graphical expression of the transition entity was specified by the property Appearance, where different graphical objects were envisaged for a transition depending if it is input or output transition. We have also added two Python coded graphical constraints to modify the appearance of the transition entity depending if the above-mentioned Boolean variables are set to .TRUE. or to .FALSE. (see Figure 1 (b)).
One of the new ideas considered in our recent developments is that the partition of the entity flows envisaged in the SC representation can be implemented by adding two Boolean variables (isCustomerOrder and isProduct) to the relations established between the net components: the arcs from a place to a transition (denoted as pl2tran components of the meta-model) and from a transition to a place (denoted as tran2pl components of the meta-model). They allow the user to specify if the current causal dependency relation models the flow of orders placed by the customer or the flow of products. Further more, different appearances are envisaged for an arc, depending if it indicates informational or material flow. We have also added two Python coded graphical constraints to modify the appearance of the relation components depending if the above-mentioned Boolean variables are set to .TRUE. or to .FALSE (see Table 2 in Section 5).

The tool generated in AToM^3 according to the above meta-specifications can be used to build, load and save refined representations of PN models proposed in our approach.
Current developments are aimed to enrich this functionality by addressing compound models synthesis. In this respect, our recent work was focused on the description at a meta-level of the rules underlying the handling of the routing constructs (RCs) identified by our method by means of models in the graph grammar formalism.

4. GRAPH TRANSFORMATIONS

Graph transformations have been studied and applied in many fields of computer science. Their wide applicability is due to the fact that graphs are a very natural way of explaining complex situations on an intuitive level. Like the token game for PN, graph transformation brings dynamic to a graphical structure, since it can describe its evolution. In particular, graph rewriting is promising as a comprehensive framework in which the transformations of complex structures can be modelled and studied in a uniform way.
The main idea of graph transformation is the rule-based modification of graphs as shown in Figure 2.

A *graph production* $p = L \xleftarrow{l} K \xrightarrow{r} R$ consists of graphs L, K and R and (injective) graph morphisms l: $K \rightarrow L$ and r: $K \rightarrow R$ mapping the interface graph K to the left hand side L and the right hand side R respectively.

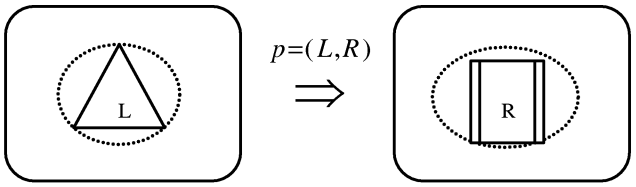


Figure 2. Rule-based Modification of Graphs

A *graph grammar* $GG = \{S, P, \Omega\}$ consists of a start graph S, a set P of graph productions and a pair of colour sets $\Omega = (\Omega_E, \Omega_V)$, where Ω_E and Ω_V are fixed sets of colours for edges and vertices respectively, and S and the graphs in P are labelled over Ω.
A graph grammar without a distinguished start graph is called *graph transformation system.*
Given a graph production $p = L \xleftarrow{l} K \xrightarrow{r} R$, a graph G and a graph morphism m: $L \rightarrow G$, called *match* of L in G, then it would be possible to define a *graph transformation* (*direct derivation*) $G \Rightarrow H$ via (p,m) if a *double-pushout diagram* (DPO) as shown in Figure 3 can be constructed.

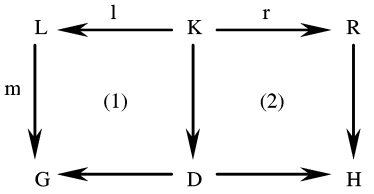


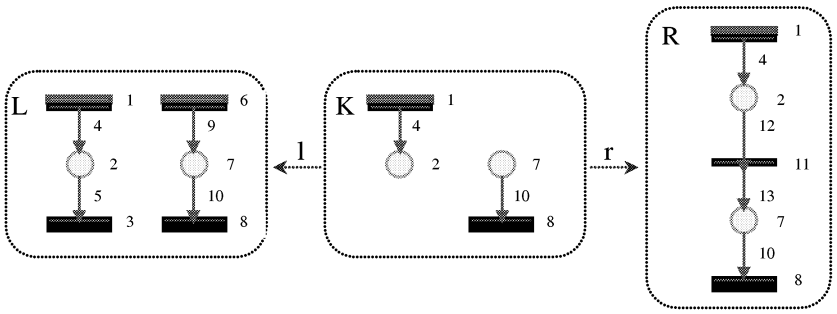
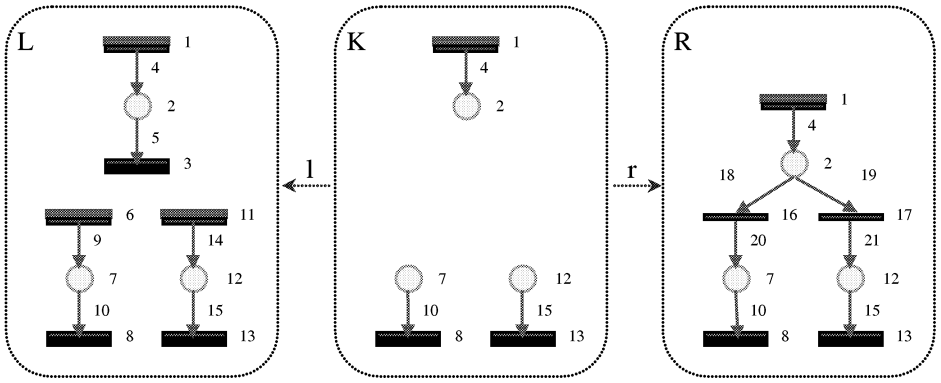
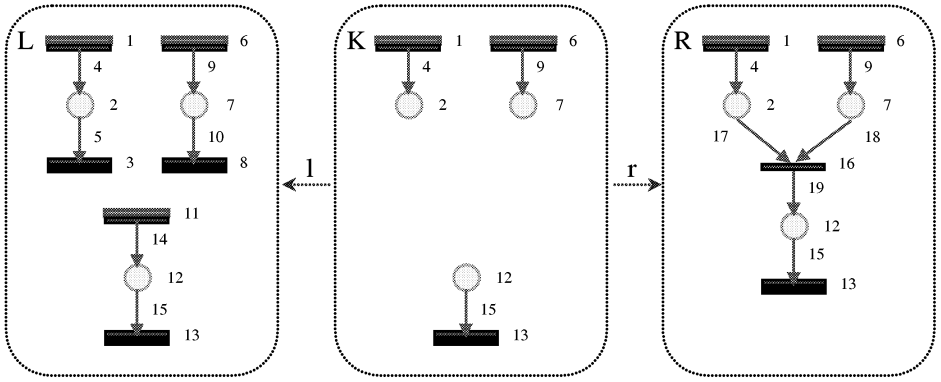
Figure 3. Graph Transformation with Pushouts

The above pushouts can be interpreted as gluing constructions as follows: Given a production p and a match m: $L \rightarrow G$, it is required to be able to construct a context graph D such that G is the gluing of L and D along K in pushout (1) and H is the gluing of R and D along K in pushout (2) in Figure 3, and it is written

$$G = L +_K D \quad \text{and} \quad H = R +_K D.$$

However, graph D (also called the *pushout complement*) may not exist, or may not be unique up to an isomorphism. In (Ehrig and Padberg 2004) it is shown that a pushout complement can be constructed if and only if a gluing condition is satisfied that requires that the boundary of the match m: $L \rightarrow G$ is included in the gluing part $l(K)$ of L.
The idea of the gluing construction of graphs makes sense also for other kinds of structures, where the union of structures along a common substructure is aimed.
In terms of Place/Transition nets, rules are introduced corresponding to graph productions in the DPO-approach. Rules describe the replacement of the left-hand side net by the right-hand side net in the presence of an interface net.

Table 1 . Rules of the Handling RC Graph Grammar

Rule 1: Sequence	
Rule 4: Conflict	
Rule 7: Synchronisation	

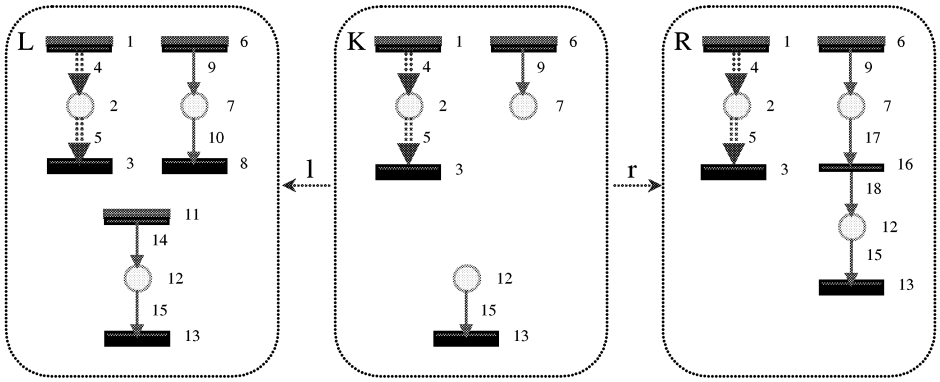
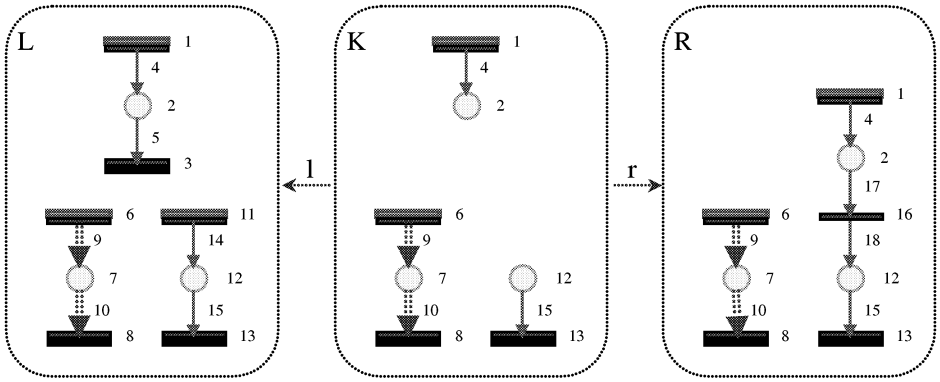
5. RULE-BASED CONSTRUCTION OF MULTICOMPONET PN MODELS

We will discuss in the sequel the rule-based construction of compound models considered in our research. The coupling step in the step-by-step procedure proposed in (Bobeanu et al. 2004) has been implemented in two different graph grammars:

1. The first graph grammar deals with the description at the meta-level of how to generate compound models by handling the RCs identified in our approach. The above-mentioned set of routings includes sequence, waterfall, concurrency, conflict, parallelism and synchronisation. Table 1 shows some of the rules of this graph grammar (the implementation is composed of seven rules).
2. The second graph grammar addresses the interfacing questions of different flows of entities. In Table 2 we show the rules allowing the implementation of the refined internal coupling templates of two sequential models while dealing with the partition of the entity flows involved in SC.

The final goal of our developments is to make the modelling process of a complex system automated from the appropriate graph grammars. At a certain extent this process will be made invisible to the user, who could be only involved in specifying the desired sequence of model manipulations.

Table 2. Rules of the Partitioned Entity Flows Graph Grammar

Rule 8: Sequence with Distinct Flows as Input	
Rule 9: Sequence with Distinct Flows as Output	

CONCLUSIONS

In this paper the combination of meta-modelling and graph transformation to support visual modelling techniques is pleaded for. Special emphasis is put on the benefits of using the information in this meta-layer of modelling in the authors' current developments of a customised tool supporting human modellers in the systematic construction of complex PN-based models.

The ultimate goal of the developments addressed in this paper is to support the automated generation of multicomponent PN-based models. The advantages of using an automated tool to construct complex models in the PN domain according to the authors' method is clear: instead of building a complex PN-model from scratch, it is only necessary to identify the RCs addressed in the real system and specify the order they should be dealt with.

REFERENCES

Banks, J. 2001. "The Future of Simulation". In: Dolgui, A., Vernadat, F. (Eds.) Conception, analyse et gestion des systèmes industriels, Volume 1, pp. 11-12. *Actes de la troisième Conférence Francophone de Modélisation et Simulation (MOSIM'01)*, Troyes, France, 25-27 avr.

Bobeanu, C.V. 2000. Contributions to Modelling and Simulation of Discrete Event Systems and Applications. Ph.D. Thesis. University of Bucharest.

Bobeanu, C.V. and Alla, H. 1998. "Modelling Basic Discrete Event Systems - A Key Factor in Supporting the Representation and Analysis of Complex Production Systems". In: Z. Binder, B.E. Hirsch and L.M. Aguilera (Eds.), Management and Control of Production and Logistics. Proceedings of the IFAC /IFIP Conference MCPL'97, vol. 1, pp. 193-198, Campinas -SP-Brazil, Aug. 31 - Sep. 3, 1997. Oxford: Pergamon Publ.

Bobeanu, C.V., Kerckhoffs, E.J.H. and Van Landeghem, H. 2004. "Formal Modelling of Discrete Event Systems: A Holistic and Incremental Approach using Petri Nets". In*: ACM Transactions on Modeling and Computer Simulation*, Vol. 14, No. 4, October. (to be published)

Ehrig, H. and Padberg, J. 2004. "Graph Grammars and Petri Net Transformations". In: Desel, J., Reisig, W. and Rozenberg, G. (Eds.). *Lectures in Concurrency and Petri Nets. Advances in Petri Nets. Lecture Notes in Computer Science*, 3098, pp. 496-536, Springer.

Kerckhoffs, E.J.H. and Bobeanu, C.V. 2001. "Systematic Structural Modelling of Discrete Event Systems using Petri Nets". In: Giambiasi, N. and Frydman, C. (Eds.) *Proceedings of the 13th European Simulation Symposium Simulation in Industry, ESS'01,* pp. 949-958, Marseille, France, SCS International Publ.

Van Landeghem, R. and Bobeanu, C.V. 2002. "Formal Modelling of Supply Chain: An Incremental Approach using Petri Nets". In: Verbraeck, A. and Krug, W. (Eds.). *Proceedings of the 14th European Simulation Symposium "Simulation in Industry",* pp 323-327, Dresden, Oct. 23-26, SCS International Publ.

Van Landeghem, H. and Bobeanu, C.V. 2003. "Using Meta-modelling to Process Petri Nets Models of Supply Chains". In: Di Martino, B., Yang, L.T. and Bobeanu, C.V. (Eds.), *Proceedings of the 2003 European Simulation and Modelling Conference, ESM2003,* pp. 513-518, Naples, Italy, EUROSIS-ETI Publ.

Zeigler, B. P., Praehofer, H. and T. G. Kim. 2000. *Theory of Modeling and Simulation*, 2nd Edition, Academic Pres.

GENERATING COLOURED PETRI NETS OF CONCURRENT OBJECT-ORIENTED PROGRAMS

Cécile Bui Thanh
LACL – Université Paris 12
61, avenue du général de Gaulle
94010 Créteil, France
bui@univ-paris12.fr

KEYWORDS
Object-orientation, coloured Petri nets, xml, net generation.

ABSTRACT

In this paper, we present a coloured Petri net generator for object-oriented modelings. The input of the generator is a concurrent object-oriented modelling described in terms of a Basic Object Oriented Notation (BOON) and provides an XML file containing the corresponding coloured Petri net. This opens a way for the simulation and verification of object-oriented modelings with any coloured Petri net based tool.

INTRODUCTION

In this paper, we present NetGenOM, a coloured Petri net generator for object-oriented modelings.

The tool works with Basic Object-Oriented Notation (BOON) (Bui Thanh and Klaudel, 2004 & 2003), an extension of a high-level programming language, $B(PN)^2$ (Best and Hopkins, 1993), which comprises most traditional concepts of parallel programming, like parallel composition, iteration, guarded commands, procedures and communications. Like usual object-oriented programming languages, BOON allows for defining classes and objects, and comprises encapsulation features, single class inheritance, polymorphism and dynamic binding. Thanks to its simplicity, it can easily be used as a basis for various extensions and the results found for it may then be applied to "real-life" languages.

The advantage of BOON is that it has a concurrent formal semantics in terms of a class of high-level (coloured) Petri nets, called *M-nets* (Best et al., 1998). The particularity of M-nets is that they are provided with a set of composition operations and allow to represent large (possibly infinite) systems in a compact and structured way. NetGenOM uses the semantics defined for BOON in (Bui Thanh and Klaudel, 2004 & 2003), in order to generate a high level petri net under an XML file. This is indeed a first step for allowing the simulation and the automatic verification of object-oriented modelings with any coloured-based Petri-net Tool.

The two next sections give the syntax of BOON supported by NetGenOM and a quick description of M-nets and the semantics of a BOON program. Then, we will describe the architecture of the tool its features, and give an overview of the format of the generated file.

THE SUPPORTED SYNTAX

We now describe the syntax supported by NetGenOM. It is somehow more complete than the syntax presented in (Bui Thanh and Klaudel, 2004 & 2003).

All names in BOON are of two kinds, denoted in the following by *classname* for class names and *name* for other names: variable names, class field names. These two kinds of names are distinguished by the first character, which is an uppercase alphabetic character for a *classname* and a downcase alphabetic character for a *name*. The remaining part of both kinds of names may contain alphabetic characters, numbers or the underscore '_' character.

Also, since the program will be associated to a coloured Petri net, standard variables (a non-object variable) must be declared with an enumerated type, in order to avoid infinite place types in the resulting net. Thus, the type of a standard variable may be built with integers, boolean or string values. It may be a set of values *set* or a range of integer *intrange*. For instance, a *set* type may be {1,2,3,4}, {"Apple","Banana","Orange"} or

{True, False}. An *intrange* type may be {1..7}.

Let $\mathbb{C}$ be the set of all defined classes. The following table gives the syntax of BOON supported by NetGenOM:

Table 1: Syntax of BOON

```
program ::= cblock block

cblock  ::= cdecl | cdecl cblock
cdecl   ::=   class classname { cscope }
            | class classname: classname {cscope}
cscope  ::=   encap fdecl
            | encap fdecl cscope
encap   ::= private | protected | public
fdecl   ::=   att name: stdtype;
            | oatt name: classname;
            | mdecl
mdecl   ::=   method name block
            | method name (parlist) block
parlist ::= param | param , parlist
param   ::=   name: stdtype | name: class-name
            | res name: stdtype | ref name:
                stdtype
stdtype ::= set | intrange

block   ::= begin scope end
scope   ::= decl scope | com
decl    ::=   var name: stdtype
            | ovar name: classname
com     ::=   <cexpr>       | meth-call
            | dotexpr.del   | block
            | com || com    | com; com
            | do alt-set od
alt-set ::=   com; repeat   | com; exit
            | alt-set [] alt-set

expr    ::=   'dotexpr   | dotexpr'   | const
            | new_C(arglist)  for all C ∈ ℂ
            | expr binop expr   | unop expr
            | (expr)
binop   ::=   +   |  -   | *   | div   | mod
            | =   | /=   | <   | >   | <=   | >=
            | and   | or
unop    ::=  -   | not
const   ::= null  | string | integer | boolean
dotexpr ::=   dexpr | this | this.dexpr
            | super.supexpr
supexpr ::= super.supexpr | dexpr
obj     ::= name | name.dexpr

mcall   ::= dotexpr   | dotexpr(arglist)
arglist ::=   aexpr   | dotexpr   | aexpr,arglist
            | dotexpr,arglist
aexpr   ::=   dotexpr | const
            | new_C(arglist)  for all C ∈ ℂ
            | aexpr binop aexpr   | unop aexpr
            | (aexpr)
```

Note that this syntax supports different standard parameter modes in method parameters. In fact, the original semantics given for BOON in (Bui Thanh and Klaudel, 2004) assumed all parameters to be passed by reference. In NetGenOM, standard parameters may be passed by value, by result or by reference, whereas object parameters are all passed by reference.

Also, NetGenOM forbids field name *overloading*. Thus, two fields owned by the same class must have different names. Nevertheless, *overriding* of inherited methods is allowed, in order to keep polymorphic aspect of BOON as an object-oriented language. In this case, the overriding method must follow the signature of the overridden method. All other constructs follow the definitions of (Bui Thanh and Klaudel, 2004 & 2003).

M-NETS AND BOON

We give here a short introduction to M-nets, and describe the semantics of BOON in terms of M-nets. More details can be found in the cited papers.

M-nets are a class of high-level Petri nets which are provided with a set of composition operations, including sequential, parallel, choice and iteration operations, and a CCS-like synchronisation called *scoping*. The particularity of the scoping is that it is a multi-way synchronisation. Indeed, transitions of M-net may are labelled with multisets of actions, and the scoping w.r.t. a set of action names synchronises complementary actions with no restriction on the number of transitions involved in the communication. The figure 1 gives an example of a scoping involving 3 transitions. At the synchronisation, the action parameters are unified in accordance with the guard, and the resulting transition is labelled by a not represented silent action.

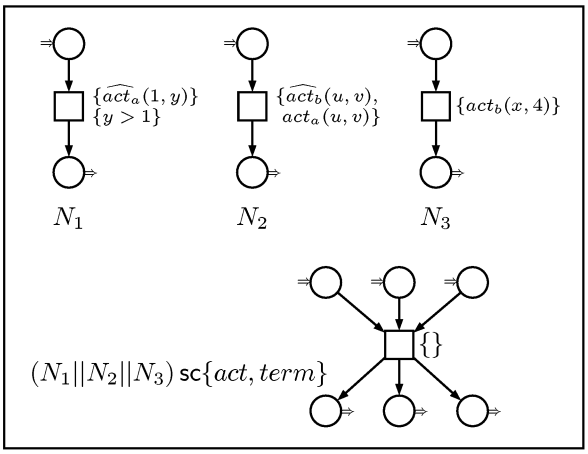


Figure 1: An Example of Scoping.

Thanks to the modularity of M-nets and the power of the scoping operation, a completely modular semantics of BOON in terms of M-nets was given in (Bui Thanh and Klaudel, 2004). These definitions can also be followed as a pattern for object oriented modelling by using directly M-nets, which are only different from coloured Petri nets by their composition operations. In fact, each M-net is a multi-labelled coloured Petri

net, which implies that after applying every composition operations, the resulting M-net is still a multi-labelled coloured Petri net, and may be analysed by any coloured Petri net based tool.

The semantics of a BOON programm is composed of M-nets which are for the most only put in parallel, and scoped w.r.t. all actions involved in the semantics. Here is an overview of the structure of the semantics of a BOON program:

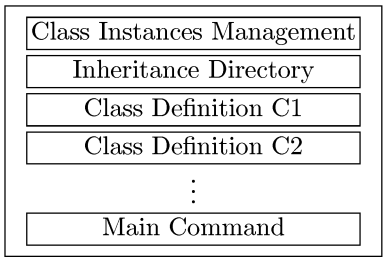


Figure 2: Structure of the Semantics of a BOON Program

Each class is represented by its own M-net, which is itself composed by several nets representing its fields. The M-net of each field contains in particular a ressource M-net, which is devoted to keep the value of each instance of the class, if the field is an attribute, or to allow the concurrent execution of the methods on all instances of the class. Objects are memorised by a couple (id, κ) where id is a unique identifier owned by the object and κ is its true type. The identifier of an object may be considered as a pointer, and is a key for retrieving its own field. The type of the objet is needed for the purpose of dynamic binding handling.

Also, each ressource M-net is associated to a field declaration. This means that an method overridden in one or more subclasses will own a ressource M-net for each new declaration appearing in the subclasses. The one to be accessed at the execution time is the one declared in the closest class to the class of the owner object. The binding of these resource M-nets to an object during the execution is handled by a couple of M-nets, called interface M-nets and Request M-nets of the corresponding field. All accesses to a field are done through its requests and interface M-nets. The access requests are forwarded by these nets to the right field ressource M-net, which is determined statically before the net generation and dynamically at the execution time by a binding of the effective class of the object variable or attribute containing the owner object. See also the figure 3. With respect to the description given in (Bui Thanh and Klaudel, 2004 & 2003), the figure 3 shows an additional *Control M-net* in charge of the initialisation and termination of the whole class M-net. That is, this net triggers the synchronous initialisation of the M-nets of the class M-net, and the synchronous termination of these nets.

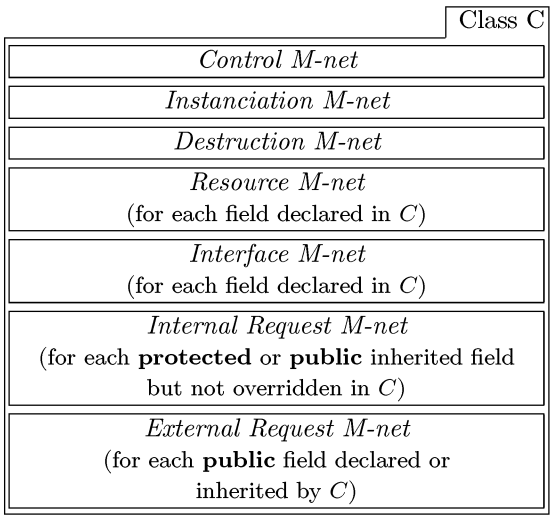


Figure 3: Structure of the M-net of a Class.

PRESENTATION OF NetGenOM

Architecture of the Tool

NetGenOM works with another tool called SNAKES (Pommereau, 2004), which is a Net Algebra Kit supporting M-net compositions. Thus, all generated nets are composed through the SNAKEStool.

NetGenOM runs under a Java Runtime Environment and needs a Python interpreter for the execution of SNAKES. This makes NetGenOMportable to all platforms supporting Java and Python. The figure 2 depicts the architecture of NetGenOM.

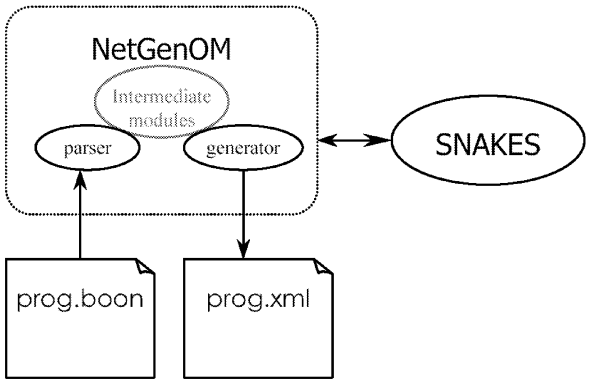


Figure 4: Architecture of NetGenOM.

The tool parses a file containing the BOON programm and prompts the program errors. If no syntax or semantic errors are found, it generates the net corresponding to the program in an XML file. The generated net is the parallel composition of all autonomous nets of the semantics, scoped w.r.t. all actions.

The Generated File Format

Discussions and works have lead the Petri net tool developpers to the development of a standard interchange format called PNML (Billington et al., 2003) and based on the XML standard. These efforts will facilitate the usability of NetGenOM outputs in other coloured Petri net tools like PEP (Grahlmann and Best, 1996), Maria (Mäkelä, 1999), or CPN Tools (Beaudouin-Lafon et al., 2001)

The output of NetGenOM is a basic format containing the description of places, transitions and arcs of the nets. Here are three exemples of outputs:

```
<place id="p1">
  <name>
    <value>
      init
    </value>
  </name>
  <type style="SET">
    <value>
      'dot'
    </value>
  </type>
  <status>
    <value>
      internal
    </value>
  </status>
  <graphic>
    <position dx="5" dy="5"/>
  </graphic>
</place>

<transition id="t">
  <name>
    <value>
      read
    </value>
  </name>
  <guard>
    <value>
      True
    </value>
  </guard>
  <actions>
    <value>
      access_V3_x!(x0r,x0w)
    </value>
  </actions>
  <graphic>
    <position dx="5" dy="7"/>
  </graphic>
</transition>

<arc source="p1" target="t">
  <label>
    <value>
      'dot'
    </value>
  </label>
</arc>
```

Note that after applying the scoping w.r.t. all actions, there won't be any actions left on the transitions labels of the net. By skipping the status of the places (needed for the purpose of composition of M-nets), the result can be easily converted to any other PNML format.

CONCLUSION AND FUTURE WORKS

We presented the Net Generator for Object-Oriented Modelling which provides a coloured Petri net XML description for a BOON program. The generated file may be exploited for the simulation or automatic verification of the modelled system, with any coloured Petri net based tool. In fact, the XML format offers many facilities for such translations. Thus, the release of translators of the NetGenOM output for some tools like are planned for the next few months. Another planned implementation is a simulator for Object-oriented modelling and modular Petri nets which would take advantage of the modularity of M-nets for allowing as well the design with Petri nets or the design with BOON.

References

[Beaudouin-Lafon et al., 2001] Beaudouin-Lafon et al. (2001). Cpn/tools: A tool for editing and simulating coloured petri nets. In *Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2001*, volume 2031 of *LNCS*. Springer.

[Best et al., 1998] Best, E., Fraczak, W., Hopkins, R. P., Klaudel, H., and Pelz, E. (1998). M-nets: an algebra of high-level petri nets, with an application to the semantics of concurrent programming languages. *Acta Informatica*, 35.

[Best and Hopkins, 1993] Best, E. and Hopkins, R. P. (1993). B(PN)^2 - a basic petri net programming notation. In *Proceedings of the 5th International PARLE, Conference on Parallel Architectures and Languages Europe*, pages 379–390. Springer-Verlag.

[Bui Thanh and Klaudel, 2003] Bui Thanh, C. and Klaudel, H. (2003). Encapsulation in an object-oriented notation based on modular petri nets. In Martino, B. D., Yang, L. T., and Bobeanu, C., editors, *Modelling and Simulation '2003, European Simulation and Modelling Conference*, Naples. EUROSIS.

[Bui Thanh and Klaudel, 2004] Bui Thanh, C. and Klaudel, H. (2004). Object-oriented modelling with high-level modular petri nets. In Boiten, E., Derrick, J., and Smith, G., editors, *Integrated Formal Methods: 4th International Conference, IFM 2004*, volume 2999/2004

of *LNCS*, pages 287–306, Canterbury. Springer.

[Grahlmann and Best, 1996] Grahlmann, B. and Best, E. (1996). Pep – more than a petri net tool. In Margaria, T. and Yi, W., editors, *Tools and Algorithms for the Construction and Analysis of Systems, TACAS 1996*, volume 1055 of *LNCS*. Springer.

[Jonathan Billington et al., 2003] Jonathan Billington and Søren Christensen and Kees van Hee and Ekkart Kindler (2003). The petri net markup language: Concepts, technology, and tools. In van der Aalst, W. and Best, E., editors, *International Conference on Application and Theory of Petri Nets, ICATPN 2003*, volume 2679 of *LNCS*. Springer.

[Jüngler et al., 2000] Jüngler, M., Kindler, E., and Weber, M. (2000). Towards a generic interchange format for petri nets. In *XML/SGML based Interchange Formats for Petri Nets, ICATPN 2000*.

[Pommereau, 2004] Pommereau, F. (2004). Snakes. Technical Report (to appear), LACL.

AUTHOR BIOGRAPHY

CECILE BUI THANH obtained a DEA/MSc in Computing Science from the University of Paris XI at Orsay in 2001. She joined then the the Laboratory of Theoretical Computer Science (LACL) of the University Paris XII as a PhD student, and supervised by Prof. Hanna Klaudel. She is working on object-oriented modelling in concurrent systems.

SIMULATION AND DETECTION OF SOME PROPERTIES OF HYBRID SYSTEM BASED DIFFERENTIAL PREDICATE TRANSITION PETRI NETS BY MEANS OF VHDL-AMS

A. Nketsa, JC. Pascal, P. Esteban
LAAS-CNRS 7 avenue du Colonel Roche 31077
Toulouse Cedex 4 France
[1] also at Université Paul Sabatier Toulouse France
E-mail : alex@laas.fr, pascal@laas.fr, gibs@laas.fr

D. Andreu
LIRMM 161 rue Ada
34192 Montpellier Cedex 5 France
andreu@lirmm.fr

KEYWORDS
Simulation, Petri nets, VHDL-AMS

ABSTRACT

This paper presents a VHDL-AMS implementation and simulation of hybrid systems based Differential Predicate-Transition Petri nets (DPT-PN). This model associates a discrete event system part described by means of Petri nets and a continuous part modelled through differential equation system. Both model are combined to give a reference one describing the behaviour of the system. Obviously formal study of this class of system is somewhat difficult, therefore simulation seems to be one of the available approaches. We use VHDL-AMS to represent both parts for simulation purposes : the system behaviour and the control. When the two models are connected, initial conditions (junction functions) for different differential equations with respect to different configurations must be satisfied. Furthermore, transitions in both models are labelled to fire at the same time. Labels are attached to sensibility functions that must be evaluated at most in one of the parts. Therefore we have used simulation to detect inconsistencies of the junctions functions and the labelling of transitions to be fired simultaneously in both models.

INTRODUCTION

The increasing resort to system integration and computer automation in industrial systems, like batch processes, has lead to deal with more and more complex hybrid system (Antsaklis & Koutsoukos, 1998). Batch processes are discontinuous systems which treat finite quantities (batches) of material of continuous nature (generally fluid or gas). The batches are submitted to an ordered set of continuous treatments using one or more pieces of equipment. The automation of these processes poses difficult issues because they combine continuous and discrete aspects. As result, both modelling and simulation of such systems cannot be easily addressed by the techniques defined for simple applications and therefore an hybrid formalism is necessary. For these purposes, we have suggested an approach for supervisory control (Pascal 2000) and defined a model, Differential Predicate Transition Petri Nets (Champagnat 1998). This model associates Petri nets and Differential algebraic equations. The places of the Petri net represent the discrete configurations of a unit. To each place, a subset of differential algebraic equations is associated (if any operation is performed in the corresponding configuration). This subset of equations describes the continuous process behaviour (chemical or physical phenomenon). In order to validate the modelling, simulation is actually the only issue. Indeed, only some works and tools are proposed in the field of hybrid system analysis, but they can only be applied to validate particular properties for special classes of hybrid systems (Villani 2004).

We propose in this paper to use VHDL-AMS to describe and simulate such system. VHDL-AMS is a well known modeling language. It allows to model both discrete event and continous systems. The latter can be described by a differential equation system. This language offers the notion of component which yield to modularity and hierarchy. This feature allows to build the top level of the system once and then building the whole system by developing the inner of the hierarchy. Since simulation is actually the best issue to study hybrid systems we deal with in this paper, this language is suitable because it allows to reproduce the flow of tokens in the net as in the formal analysis and therefore permitting to observe some properties of the Petri net.

In section 2, we introduce an overview of Differential Predicate-Transition Petri nets (DPT-PN). Section 2 deals with the modeling of hybrid systems by means of DPT-PN. An example is then given in section 4. Section 5 discusses the VHDL-AMS implementation and simulation strategies. Section 6 reviews some expected results and the final section draws conclusions and discusses some perspectives.

OVERVIEW OF DIFFRENTIAL PREDICATE TRANSITION PETRI NETS (DPT-PN)

Considering that the reader is familiar with Petri nets, the main characteristics of the DPT Petri nets are briefly presented. Taking as a starting point an ordinary Petri net, the characteristics of DPT Petri net [Champagnat 1998), (Pascal 2000) :

- A set of continuous variables (x_i) is associated with each token.
- A differential equations system (F_i) is associated with each place (P_i); it defines the dynamic of the variables x_i associated with the tokens in P_i, according to the time.

$$F_i(\dot{X}_i, X_i, \theta) = \begin{bmatrix} f_{i_1}(\dot{X}_i, X_i, \theta) = 0 \\ \vdots \\ f_{i_n}(\dot{X}_i, X_i, \theta) = 0 \end{bmatrix}$$

- An enabling function (e_i) is associated with each transition (t_i); this function defines a threshold on continuous variable x_i associated with the tokens of the input places (X_{input_i}).

$$e_i(X_{input_i}, \square) <, =, > 0$$

When the transition t_i is enabled by the presence of tokens in its input places, the enabling function is considered and the differential equation system (F_i) is solved. Then, when the enabling function becomes true, the transition is fired.

- A junction function (j_i) is associated with each transition (t_i); it defines the value of the variable x_i associated with the tokens of the output places (X_{output_i}) after the transition firing. This value corresponds to the initial value of the variable x_i used in the differential equations system (F_i) of output place.

$$X_{output_i}(\square^+) = j_i(X_{input_i}(\square^-))$$

MODELING HYBRIB SYSTEMS USING DPT-PN

For supervisory control purposes, the model must first supply a discrete view, describing the configuration in which the system is. Indeed in a batch operated plant, the process equipment is not in continuous use, units are used sporadically with changes between active, waiting and inactive status. At different instants of time, the plant is then in different configurations, each configuration corresponding to a set of active connections between units. The supervisory controller drives the production system from dynamic configurations to dynamic configurations in order to achieve the fabrication of the product.

The model must also supply a continuous view; this view describes the continuous behaviour of the system within a given configuration. The process behaviour fluctuates depending on the unit operations being processed. So then, the hybrid model associates the set of continuous system behaviours to the set of discrete views. This model, called hybrid reference model (Andreu 1998), corresponds a qualitative view of the behaviour of the units independently from any control strategy. All the states described on the reference model must be the reachable configurations and this finite set of discrete states must not be restricted by control considerations. It can only be limited by technological constraints.

Thus, a qualitative aspect is associated to the discrete states (the configurations). According to an active configuration due to the beginning of an operation, it is possible to qualitatively determines the expected continuous phenomena evolution. For instance, in a configuration corresponding to a storage tank loading, the volume must be increasing. If it is not the case, a failure is detected.

The hybrid model of the process is then composed of a set of hybrid models: one model by unit. The discrete aspect of each model is represented by means of Petri nets. The places of the Petri net correspond to the configurations of the unit in relation to its environment (connections to others units for example). With each discrete state of the net (each place), a subset of differential algebraic equations is associated (Andreu *and al.*, 1996; Valette *and al.* 1995; Champagnat *and al.* 1998). This subset of equations describes the expected continuous process behaviour. The active unit configuration is characterised by the location of the token in the net (the marking) and its continuous state, described by a subset of continuous variables, is carried by the token. The set of marked places of the nets (one net by unit) corresponds to a logical description of partial states of the whole controlled system.

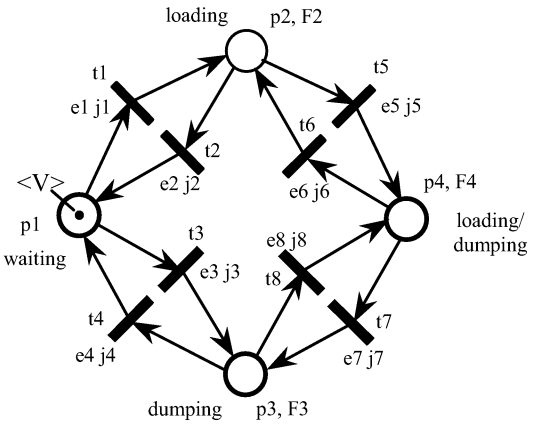
When the control system evolves from an operating stage to another one, the reference model simultaneously evolves. A transition firing in the control module (the control strategy is also described by means of Petri nets), which corresponds to the execution of a given operation, simultaneously results in the firing of the associated transition in the reference model. The simultaneous evolution of the reference model with the control one ensures that the automatically selected subset of equations is consistent with the operations in progress.

The use of the reference model has multiple objectives. The model must permit the detection of abnormal behaviours in comparison with the expected behaviour for the performed control. Thus, it is used to validate the execution of an operation. Furthermore it is sometimes useful to simulate the process behaviour on a

short time horizon. The aim is then to verify that the command to apply will not lead to constraints violation. According to this emulation principle of the reference model, it can be envisaged to analyse the consequences of a decision before making the process recovery (Andreu 1998).

EXAMPLE

Let us consider the example of a feed tank unit. This unit has four configurations: waiting, loading, dumping, loading/dumping. That corresponds to the four discrete states of the configuration model. The place p1 corresponds to the tank in waiting configuration with a current capacity V (attribute of the token). At the places p2, p3, p4 are respectively associated the equations F2, F3, F4. These equations allow to calculate the capacity variation of the tank according to the current capacity V and the input flow qi or output flow qo. At each transition ti, an enabling function ei and a junction function ji are associated.



$$F_2 = \frac{dV}{dt} - qi \qquad F_3 = \frac{dV}{dt} + qo \qquad F_3 = \frac{dV}{dt} + qo - qi$$

Enabling functions

$e1, e3, e5, e7 : VRAI$

$e2, e4, e6, e8 : V = Vspec$

Junction functions

$j1, ..., j8 \ : \ V(\theta^+) = V(\theta^-)$

Figure 1 : Reference model

Let us assume a control model that performs a loading operation and a dumping operation. At initial time, the place pa is marked by a token with attributes qi (input flow), qo (output flow) and Vspec (the capacity to reach). The following transitions are labelled to be merged with their corresponding in the reference model (figure 1) : ta (resp tb, tc and td) is labelled with t1 (resp t2, t3 and t4) (figure 2). This means that they will be fired at the same time when evolution conditions are satisfied.

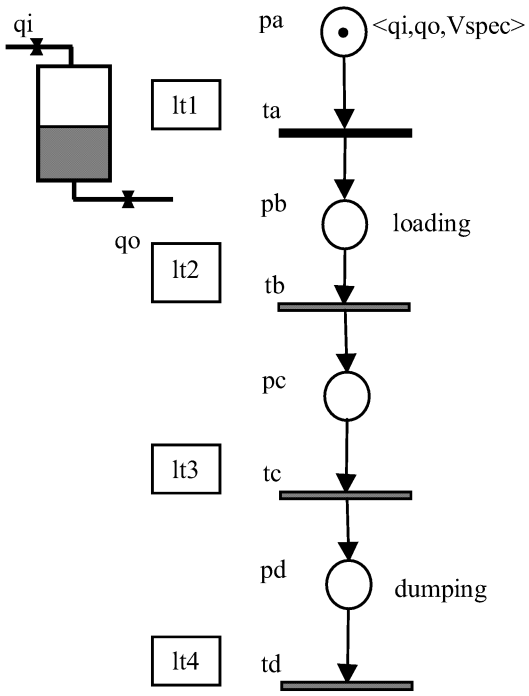


Figure 2 : control model

When the transition ta is fired, corresponding to the starting of the loading operation, the transition t1 of the configuration model is simultaneously fired because they are labelled (lt1) for merging. In the control model, the place pb is marked. In the reference model, the place p2 is marked with a token whose attributes are the current capacity V, the input flow qi and the capacity Vspec to be reached. The corresponding equation is activated in order to calculate the date of the loading-end operation. When the calculated capacity reaches the value Vspec specified by the enabling function e2 associated to the transition t2, t2 is fired. Consequently the attribute V of the token is updated by the junction function j2 with the calculated value. The place p1 becomes marked. The firing of the transition t2 simultaneously results in the firing of the enabled transition tb in the control model and to the marking of the place pc. When the dumping operation is started (transition tc firing), the configuration model evolves in the dumping state (simultaneous firing of transition t3). The place p3 is marked and the equation F3 is activated. When the capacity V reaches the new Vspec value, the transition t4 is fired and the token attributes are updated. In the control model, the transition td is simultaneously fired, corresponding to the end of the dumping operation.

IMPLEMENTING DPT-PN HYBRID SYSTEMS IN VHDL-AMS

Petri net and VHDL-AMS are well known and are frequently used tools for respectively modeling and simulation. Therefore we assume that the reader is familiar with ordinary Petri nets and the modeling language VHDL-AMS. The implementation of a DPT-PN model needs at least two sub-models : the reference model

to describe the process, and the control model. The reference model can be split in many sub-models corresponding to different parts of the process. Each sub-model is made of two parts which are merged: a Petri net model and a continuous model consisting to differential equation system. The control model which allows the reference model to toggle between process configurations is a Petri net.

To achieve this implementation in VHDL-AMS, we will use the component approach inherent to this modeling language. For the seek of clarity, we will not detail the components nor the declaration part of the whole program.

Components

The component place updates the marking of every place of the net

Clk is used for synchronous evolution of the marking. There are other kind of signals used to reset to set the initial marking, set marking,reset marking (condition of place marking). The component outputs then the new reached marking. The entity of all the place in the net is :

Place

```
generic (number_place : natural :=1);                    --
number of places in the net
Port (     clk,reset : in   std_logic;
init_marking       : in   std_logic_vector(number_place-1
downto 0);
set marking,reset marking: in
std_logic_vector(number_place-1 downto 0);
marking : out std_logic_vector(number_place-1 downto 0)
)
```

The component transition_ams is a combinatorial process which computes the evolution expressions of the marking for the reference model. We point out that, we can handle different types of transitions (i.e. classical, inhibitor and test arcs).
Vspec is the threshold, Vout is the value produced by a differential equation

Transition_ams

```
generic (number_place     : natural :=1;
          number_transit    : natural :=1);
Port (     Vspec                        : in
quantity_vector(number_transit-1   downto 0);
 Vout     : in    quantity_vector(number_place-1
          downto 0);
          transit_label : in
          std_logic_vector(number_place-1   downto 0);
           marking: in    std_logic_vector(number_place-1
                    downto 0);
           junction: out  quantity_vector(number_transit-1
                    downto 0);
           setmarking, resetmarking: out
          std_logic_vector(number_place-1   downto 0));
```

Transition is built from a generic transition template and merging labels and so on.

```
Dynamic generic( number_equation  : natural;
                 number_transition: natural;
                 number_place      : natural);
Port   (marking  : in   std_logic_vector(number_place-1
         downto 0);
junction : in   quantity_vector(number_transit-1
downto 0);
Vout: out quantity_vector(number_equation-1 downto 0)
);
```

The component dynamic is used for the description of the continuous part. Its aim is first to solve the differential equation system and second to synchronize the continuous and the discrete parts of the reference model. This synchronization is based on the VHDL-AMS instruction break which gives the opportunity to start a solver when an event (marking update in our case) is triggered and furthermore defines the initial conditions (junction function in our case) for the starting integration of differential equations.

Components for the control model

Transition_des

```
        generic (number_place     : natural :=1;
                 number_transit    : natural :=1);
        Port (     token_attribut     : out
token_attrib_vector (nber_transit-1 downto 0);
        transit_label       : in
        std_logic_vector(number_place-1   downto 0);
        marking : in    std_logic_vector(number_place-1
                 downto 0);
        set marking, reset marking            : out
        std_logic_vector(number_place-1   downto 0) );
```

The component transition_des is a combinatorial process which computes the evolution expressions of the marking for the control model.

The top level description becomes only the instanciations of these components :

TOP LEVEL DESCRIPTION

a) Top level reference model for the example described above :

```
 Place_ref : Place
  generic map (4)
  port    map (clk, reset, init_marking, setmarking ,
  resetmarking, marking);

 Transit_ref: Transition_ams
  generic map (4,8)
  Port map (  Vspec, Vout, transit_label, marking,
  junction, set_marking, reset_marking)

 Output_ref : Dynamic
  generic map (3,8,4)
  port  map  (marking, junction,   vout);
```

b) Top level control model for the example described

above :

```
Place_control : Place
  generic map (4)
  Port map (clk, reset, init_marking, setmarking ,
  resetmarking, marking);

Transition_control :Transition_des
  generic map (4,4);
  port map (token_attribut, transit_label, marking,
  setmarking, resetmarking);

Action_control : Action_place_des
  generic map (4)
  port map (token_attribut, marking);
```

SIMULATION STRAGEGIES AND EXPECTED RESULTS

The DPT-PN model needs some consistencies to be checked before evaluation. These consistencies are :
- merged transitions in reference model and control models must be labelled by the event.
- merged transitions must be associated to enabling functions in at most one of the two models.
- junction functions must not allow inconsistencies for the continuous variables.

In order to check theses consistencies, we have built the verification mechanism on the base of test-benches in VHDL-AMS. Test-bench consists in considering two parts : a VHDL system under test and another VHDL program to generate the required inputs and analyze the output produced by the system under test. This analysis can be done automatically in VHDL by using the instruction "assert" to catch different produced output configurations.
In order to check the expected consistencies, the test-bench program reproduces formal analysis by setting the conditions associated to transitions. The Petri net is run by generating a clock for marking updating. In this manner tokens flows through the places. For the third consistency about junction functions, the continuous part as been reduced to produce identified continuous values such that we can detect conflict between these variables.

CONCLUSION

We have proposed a way to study a class of model used to described a complex system. The approach is based respectively on the modeling and simulation language DPT-PN and VHDL-AMS. We have shown that this hybrid Petri net based can be handled in VHDL-AMS. Moreover the extension of this approach to a bigger model is straightforward. The simulation strategies have proved that the approach can mimic a part of the formal analysis to track some constraints prior to the evaluation of the system. As a perspective, we plan to introduce time in both modeling and simulation.

REFERENCES

Pascal JC. 2000 Modular and hierarchical approach for supervisory control of batch processes.
In The 4th International Conference on Automation of Mixed Processes: Hybrid Dynamic Systems (ADPM2000). Dortmund (Germany), 18-19 September 2000. Proceedings pp. 369-374

E. Villani, J. C. Pascal, P. E. Miyagi, R. Valette 2004 Object oriented approach for cane sugar production: modelling and analysis. *In Special issue: Analysis and Design of Hybrid Systems - Edited by S. Engell, H. Gueguen and J. Zaytoon, Control Engineering Practice, Vol 12/10 pp 1279-1289, 2004*

Antsaklis, P., Koutsoukos, X.D., & Zaytoon, J., 1998. *In On Hybrid Control of Complex System: A survey . European Journal of Automation 32 (9-10), 1023-1045.*

Champagnat R., Andreu D, Pingaud H., Pascal JC., Esteban P. (1998). Modelisation des systèmes de production discontinus : proposition d'un modèle pour la superposition. *APII Jesa vol 32 n°56 Septembre 1998 pp 725-756*

Daubas B., Pages A. & Pingaud H. 1994. *Combined Simulation, of Hybrid Processes. IEEE International Conference on SMC, p. 314-319, San-Antonio, USA, October 1994.*

Johnsson C. & Arzen K.E. 1994. *High-level Grafcet and Batch Control. ADPM'94 Symposium on Automation of Mixed Process: Dynamical Hybrid Systems. Brussels, p. 255-263, Belgium, november 1994.*

Andreu D., Pascal J.C. (1998). *An hybrid reference model for on-lined detection in discontinuous systems. IFAC/IFIP workshop on "On-line fault detection and supervision in the chemical process industries". Solaize (Lyon), France, juin 1998*

Valette R., Pingaud H., Pages A., Andreu D. & . Pascal J.C. (1995). *Modeling, Simulation and Control of Event-Driven Operations in Process Systems. ETFA'95 INRIA/IEEE Symposium on Emerging Technologies and Factory Automation, vol. 3, p. 119-128, Paris, France, october 1995.*

FLUID FLOW SIMULATION

CANDU STEAM GENERATORS MODELING USING COMPGEN FOR MMS PACKAGE

Ilie Prisecaru, Daniel Dupleac and Adrian Constantinescu
Power Plant Engineering Faculty
Politehnica University of Bucharest
313 Splaiul Independentei, R-77206, sector 6, Bucharest, Romania
E-mail: prisec@cne.pub.ro

KEYWORDS
Nuclear engineering, Model design, Verification, Lumped parameter, Continuous simulation

ABSTRACT

MMS package is an advanced tool for thermal-hydraulic analyzes. It is built on a modular philosophy that allows the user to make its own schemes utilizing the existing MMS modules. Also new MMS modules can be developed by using CompGen tool. The paper briefly describes the development of new CANDU plant modules to be used with MMS package at Nuclear Engineering Department of "Politehnica" University of Bucharest (UPB) and presents the model developed for the Steam Generators.

INTRODUCTION

The modeling of thermodynamic processes, in stationary and dynamic regimes, is very important for the design and operation of nuclear power plants. In the last years, many computing codes - that can be used on simulation analyzes - were developed and improved. One of these codes is MMS (Framatome Tehnologies 1998a). MMS package was developed during the years by Framatome Technologies, Babcock & Wilcox and nHance Technologies Inc. It uses a modular architecture and it is based on ACSL program (Mitchell and Gauthier Associates 1993). MMS is provided with a generous library of modules that includes control, electromechanical, gas, water, boundary, fossil and nuclear modules.

The application of the MMS package at the Nuclear Power Plant Department of the Faculty of Power Plant Engineering was oriented towards the (Prisecaru et al. 1996):

1. Determination of the plant performances for stationary and transitory regimes.

2. Educational simulators dedicated to entire Nuclear Power Plant or subsystems.

3. Analysis of abnormal plant operation.

4. Testing and validation of control system.

The development of CANDU models for the MMS package is being performed in order to help the Romanian designers of Cernavoda Nuclear Power Plant at thermal hydraulics analysis of auxiliary systems of Primary Heat Transport System and Balance of Plant Systems. For this purpose, models for equipment, common to different types of nuclear plants, as local air coolers and various heat exchangers were developed using the MMS module library.

Although this equipment can be modeled with existing MMS modules, there is lack of modules for specific equipment in nuclear plants in general - i.e. spent fuel bay and radiological protection - , and in particular, for a reactor type – i.e. calandria vessel and heavy water enrichment tower in CANDU plants. Moreover, there is a need to change certain characteristics of mathematical model in some modules in order to obtain a more accurate response in the transient regimes. For these, new modules were developed using CompGen package (Framatome Tehnologies 1998b) and a CANDU specific equipment library was created. CompGen package allows the user to implement in MMS frame a new model written in C language. In this paper the development of the Steam Generator model with CompGen is presented.

MODEL DESCRIPTION

The thermal behavior of the steam generators in water reactor plants plays an important role in determining the power plant response during normal and abnormal conditions. Thus modeling the steam generator performance is an important step in obtaining accurate response of the plant during the transients. Different models of steam generators may be found in literature. Generally, in the safety analysis code, for example RELAP5 (Thermal Hydraulics Group 1998), the components of the plant are modeled starting from the basic two-fluid model, which is applied, to volumes and junctions. Thus, modeling steam generators will consist in linking the number of the desired volumes to obtain the accuracy required by the analyzed transient. Although a detailed model of steam generators may be developed (La Cour Cristensen 1973; Nahavandi and Herman 1976; Hoeld 1978), in which several control volumes may be considered, in most thermal hydraulics analyses of the plant behavior only the global response of the main steam generator parameters is needed. In this case, good prediction may be obtained with simplified non-linear

models (Matausek 1974; Kerlin 1979). One such model is the MMS's RSGe module provided in the MMS library (Framatome Tehnologies 1998a).

The MMS's RSGe module simulates a natural circulation U-tube Recirculating Steam Generator with a feedwater preheater. The major drawback to apply this module for the CANDU plant consists in the lack of possibility to use heavy water on the primary steam generator side. Thus, first attempt in developing the CANDU steam generator model, namely CANDU_SG, was to use the same mathematical model like in RSGe module and to adjust it to the CANDU steam generator features. The above model uses two pressures for secondary side description: the dome region pressure (steam pressure) and the U-tube region pressure. Further on, an alternate simplified module was developed, namely CANDU_SG_S. For this module, the mathematical model uses only one pressure to describe the secondary side of the steam generators, that is the steam pressure. The main features of these models and the differences between them are described in the following.

The primary side of both models is divided in four regions that correspond to: ① condensing region of the reactor coolant at the inlet of hot leg; ② remaining part of the hot leg; ③ upper part of the cold leg, and ④ the region of cold leg that corresponds to the preheater, as shown in figure 1.

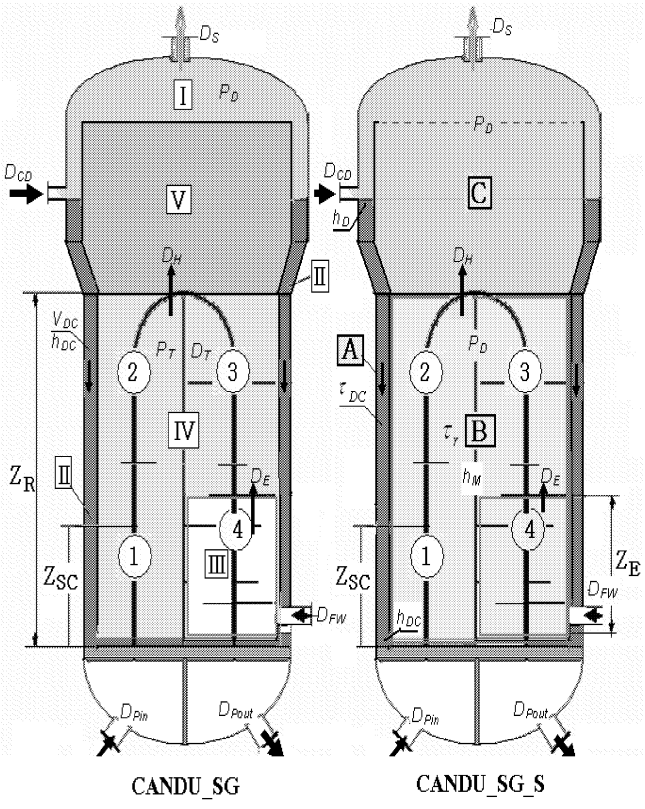


Figure 1: Steam Generator CANDU Model Description

The primary side is treated identically in both models. The inlet and outlet primary pressures are the boundary conditions and the model calculates the flow rate inside U-tube. The enthalpy variation of each primary region is calculated through the energy conservation equation. For the subcooled region of the preheater, the Dittus-Boelter correlation for the heat transfer on the primary side and a correlation for shell-side coefficients from McAdams on the secondary side are used (Frass and Ozisik 1965). In the boiling regions, the heat transfer correlation used is Dittus-Boelter on the primary side and Chen on the secondary (Collier and Thome 1994). A constant conductivity value for the U-tube wall was used. This assumption is adequate, as the wall temperature variation is small in almost all of the transient of interest.

The main differences between models arise on the secondary side of the steam generator. For both models, particular attention was paid to the distribution of mass in the secondary side during transients, given these parameters are the main contributors to the transient time constants. In the CANDU_SG model, as well as in MMS's RSGe module (Framatome Tehnologies 1998a), the secondary side of the generator was divided into five regions, marked with I to V in figure 1: steam dome, downcomer, preheater, heated riser, respectively unheated riser/separator region. The model is formulated by applying the conservation of mass and energy to each region. The particularity of this model consists in using the dome pressure at describing fluid properties for steam dome and downcomer region and the U-tube region pressure at describing fluid properties for preheater, heated riser and unheated riser/separator region. In this way, the steam generator is practically divided in two regions of same pressure in which the conservation equation written for each of the five lumped regions described above can be coupled in terms of pressure, flow and/or enthalpy. Thus, coupling the mass and energy equations for regions with same pressure gives rise to a three unknown system with three equations, which may be written in matrix form as:

$$\|A\| \cdot |X| = |C| \qquad (1)$$

where, for the steam dome and downcomer region, the state vector X is given by the time derivatives of steam pressure, downcomer water volume and downcomer enthalpy; for the preheater, heated riser and unheated riser/separator region, the state vector X is given by the time derivative of the U-tube region pressure, exit preheater flow rate and exit riser flow rate.

From the mass and energy equations, the time derivative of the exit preheater enthalpy, exit riser and unheated riser-separator quality are also found in terms of X state vector variable. Two momentum equations are used on the secondary side for the calculation of the recirculation flow. The unknowns of these two momentum equations are the flow out of the downcomer into the boiling region and the flow leaving the unheated riser/separator region. These variables are also state variables, thus the model has eleven

state variables on the secondary side and four on the primary side. A fourth order Runge-Kutta method was chosen for solving the time dependent model equations (Mitchell and Gauthier Associates 1993).

In the CANDU_SG_S model all fluid properties are calculated for the steam dome pressure and the model is further simplified. From the energy point of view, the downcomer is characterized only by a time constant; thus, the energy balance equation is not used for this region. The downcomer leaving enthalpy is actually the dome water enthalpy delayed with the downcomer transport time. For the inlet riser enthalpy, a perfect mixing between downcomer flow and the preheater flow is assumed. Thereafter, the exit riser enthalpy is computed in quasi-static mode and is delayed with the riser flow transport time. The exit riser quality is computed starting from the delayed exit riser enthalpy and the unheated riser quality is considered constant and equal to the exit riser quality. The secondary average void fraction, needed for natural circulation head, is computed as volume averaged between the average riser void fraction and the unheated region void fraction. Then, the recirculation flow is obtained by equating the natural circulation pressure head with the pressure losses around the natural circulation loop formed by downcomer, riser and steam drum. Thus, only one momentum equation written for steady state case is used.

The steam pressure and the volume of water in the steam dome are obtained through the equations of liquid and vapor phase mass conservation and energy conservation for the steam dome. This leads to one differential equation for the steam dome water volume (thus the water level in the steam generator), one differential equation for the steam dome density and one differential equation for the steam dome water enthalpy. Through steam property functions, the steam pressure is found starting from the dome steam density. Details of this model can be found in (Mensah 1982).

Steam and water properties for light water included in the MMS library was used, while heavy water properties have been developed previously (Prisecaru et al. 2001).

The user must input the initial values of steam generators main parameters (i.e. coolant flow rate, inlet coolant enthalpy, inlet and outlet coolant pressure, feed water flow rate and enthalpy, reheater drains flow rate and enthalpy, steam pressure, water level and circulation ratio) and geometrical data. Consequently, the auto parameterize process calculates the steam flow rate, steam generator power, recirculation flow rate, outlet coolant enthalpy and the distribution of mass in secondary steam generator regions. The steam generator model has two linking ports for the primary side (inlet and outlet of the reactor coolant) and three linking ports for the secondary side (steam, feedwater and reheater drains). The primary side linking ports are of the resistive type, namely the module compute the primary coolant flow rate given the inlet and outlet coolant pressure. The secondary side linking ports are of storage type; the module computes the pressure given the flow rate at the inlet and outlet of the volume. The block diagram, representing the entry and the leaving module variables, is identically for both modules and is given in figure 2.

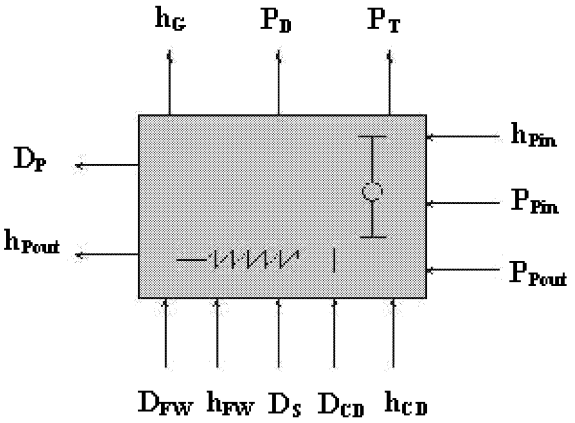


Figure 2: Block Diagram

RESULTS

A validation process for the two steam generator models developed are made. The validation aims at finding whether the mathematical model was correctly accomplished within modules. A perturbation test was chosen in order to verify the steam generator modules, in which CANDU_SG and CANDU_SG_S response to perturbation were compared with the MMS's RSGe module. For this purpose, the ports of the modules were linked through the pipe components to the boundary modules and a control valve was used to control the water level, as shown in figure 3. This arrangement is a good approximation of the main steam and feedwater systems of CANDU plant. A 20% steam generator power reduction transient starting from full power condition was simulated applying a ramp coolant enthalpy decrease between 300 s and 600 s transient time.

The ramp perturbation in the inlet coolant enthalpy, figure 4, decreases the heat transferred in the steam generators. Hence, the exit riser enthalpy and quality decreases. These lead to a slight decrease of the steam pressure and, consequently, to the decrease of steam flow. Therefore, the feed water flow overcomes the steam flow; as a result, the water level raises and the control valve closes to decrease the incoming feed water flow. Comparison of the obtained results for the main steam generator parameters is shown in figure 5 for steam pressure, figure 6 for steam flow rate and figure 7 for water level transient response. As shown in the figures 4 to 7 the response of created modules is practically identical with the response of MMS's RSGe module, both qualitatively and quantitatively. In fact, the CANDU_SG and CANDU_SG_S results are the same. The slight differences with the results of the RGSe module are mainly due to the initial instabilities in steady stade regime.

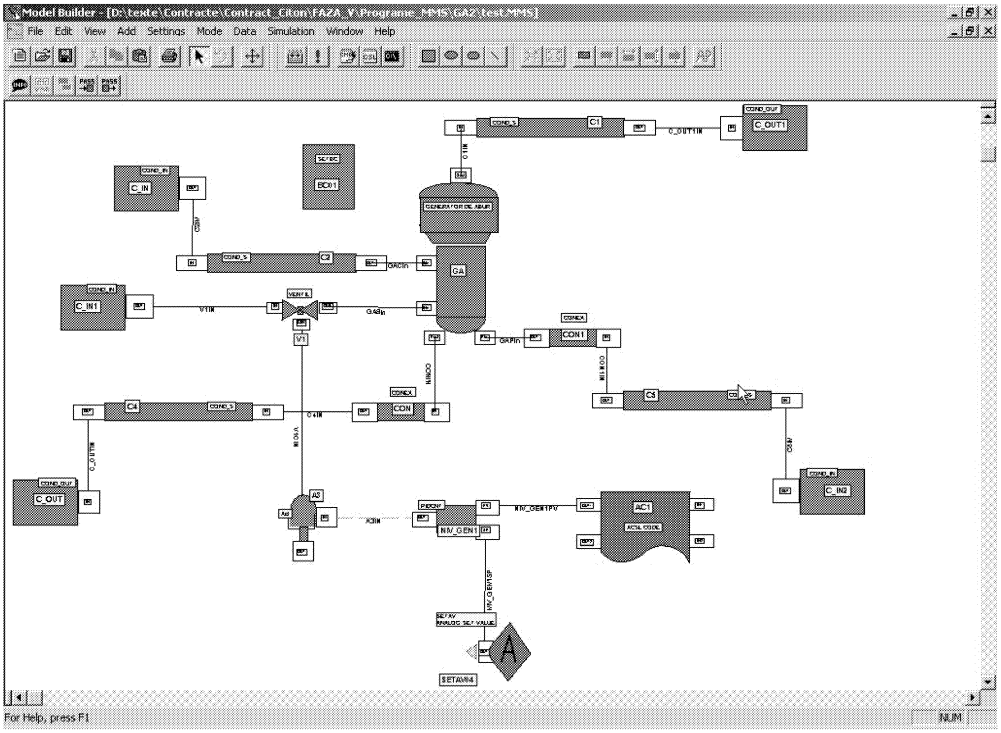


Figure 3: CANDU_SG_S Model Test

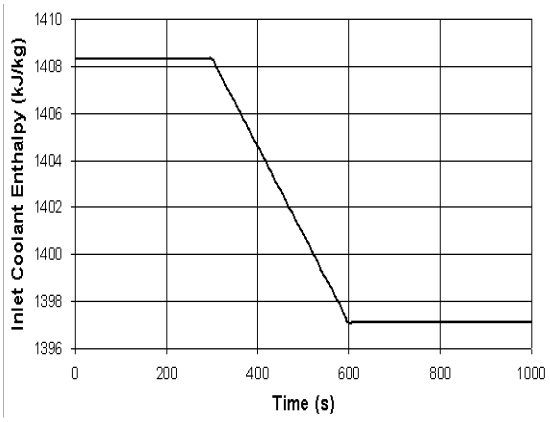


Figure 4: Inlet Coolant Enthalpy Ramp Reduction

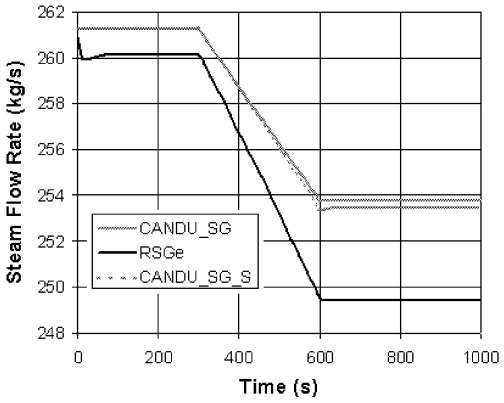


Figure 6: Steam Flow Rate Transient Response

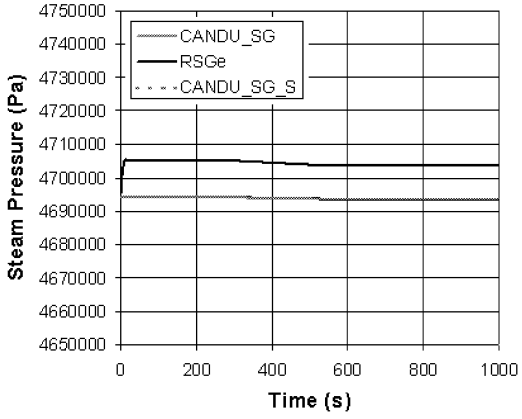


Figure 5: Steam Pressure Transient Response

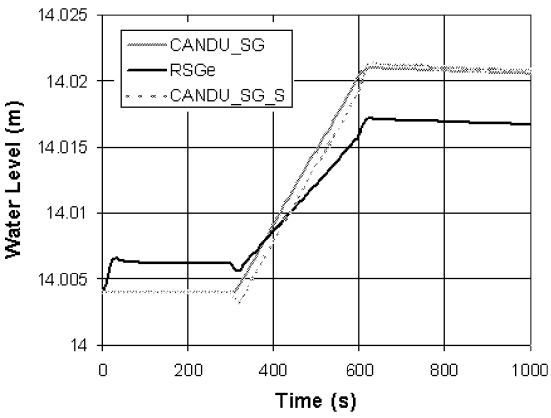


Figure 7: Water Level Transient Response

CONCLUSIONS

New CANDU plant modules, to be used with MMS package, were developed at the Nuclear Engineering Department of UPB. The aim was to help the Romanian designers of Cernavoda NPP at performing thermal hydraulics analysis of auxiliary systems of Primary Heat Transport System and Balance of Plant Systems. For the steam generator, two models were developed, namely CANDU_SG and CANDU_SG_S. The differences between these models derive from different approach in which the secondary side was treated. The models were implemented using CompGen in the MMS package. The results presented in the paper for the 20% steam generator power reduction transient starting from full power condition demonstrate that the CANDU steam generators modules have been properly developed. Moreover, these modules are more stable in the steady stade after the auto parameterize process than the MMS's RSGe module that, at the initial running time, drift from the steady stade input values. The new modules were inserted in the CANDU MMS library to be used in thermal hydraulic analyses.

REFERENCES

Collier, J.G.; and J.R.Thome. 1994. *Convective Boiling and Condensation*, Clarendon Press.

Framatome Tehnologies. 1998a. *MMS Theory Manual Release 5.1*.

Framatome Tehnologies. 1998b. *CompGen Users Manual*.

Frass, A.P.; and M.N. Ozisik. 1965. Heat Exchanger Design, John Wiley & Sons, Inc., New York, London, Sydney.

Hoeld, A. 1978. "A Theoretical Model for the Calculation of Large Transient in Nuclear Natural Circulation U Tube Steam Generator." *Nuclear Engineering and Design*, No.1, 1-23.

Kerlin, T.G.; E.M. Katz; J.D. Freels; and J.G. Thakkar. 1979. "Dynamic Modeling of Nuclear Steam Generators". *Proceedings of 2nd Int. Conference on Boiler Dynamics and Control in Nuclear Power Station* (Bournemouth). 249-258.

La Cour Christensen, P. 1973. "Description of a Model of a U-Tube Steam Generator" Technical report R150-M-1564 (Jan.)

Matausek, M.R. 1974 "SINOD-A Nonlinear Lumped-Parameter Model for Steady-Stade, Transient and Stability Analysis of Two-Phase Flow in Natural Circulation Boiling Water Loops." *Nuclear Science and Engineering.* No. 53, 440-457.

Mensah, S. 1982. " Multivariable controller for 600 MWe CANDU Nuclear Power Plant." AECL Report, No. 7841F.

Mitchell and Gauthier Associates (MGA) Inc. 1993. *Advanced Continuous Simulation Language (ACSL) Reference Manual.* Edition 10.1

Nahavandi, A.N.; and S. Herman. 1976. "Dynamic Analysis of Vertical U-Tube Natural Circulation Steam Generators". *Proceedings of the Two-Phase Flow and Heat transfer Symposium-Workshop* (Fort Lauderdale, Florida, Oct.18-20).

Prisecaru, I.; M. Constantinescu; and D. Dupleac. 1996. "Towards an Integrated Strategy for Modeling and Simulating the Transitory Regimes in NPP. A Proactive Educational and Training Oriented Approach at the University Politehnica of Bucharest." *Proceedings of The National Energy Conference* (CNE'96, Neptun, Romania, 1-5 Sept.), Section IV, 163 –170.

Prisecaru, I.; D. Dupleac; and A. Constantinescu. 2001. "personal communication."

Thermal Hydraulics Group. 1998. "RELAP5/MOD3 Code Manual Volume 1: Code Structure, System Models, and Solution Methods" Scientech, Inc., NUREG/CR-5535.

NOTATION

D_{CD} – reheater drains flow rate, kg/s
D_E – exit preheater flow rate, kg/s
D_{FW} – fedd water flow rate, kg/s
D_E – exit heated riser flow rate, kg/s
D_{Pin} – inlet primary side flow rate, kg/s
D_{Pout} – outlet primary side flow rate, kg/s
D_S – steam flow rate, kg/s
D_T – riser flow rate, kg/s
h_{CD} – reheater drains enthalpy, kJ/kg
h_{DC} – downcomer enthalpy, kJ/kg
h_G – steam enthalpy, kJ/kg
h_{FW} – feedwater enthalpy, kJ/kg
h_M – mixing region enthalpy, kJ/kg
h_{Pin} – inlet primary side enthalpy, kJ/kg
h_{Pout} – outlet primary side enthalpy, kJ/kg
P_D – steam pressure, Pa
P_T – tube bundle region pressure, Pa
V_{DC} – volume of water in downcomer, m^3
Z_E – preheater lenght, m
Z_R – riser lenght, m
Z_E – subcooled preheater lenght, m
τ_{DC} – downcomer transport time, s
τ_r – riser transport time, s

AUTHORS BIOGRAPHY

ILIE PRISECARU was born in 1947 in Bucharest, Romania. He is the head of the Nuclear Power Department at Faculty of Power Plant Engineering of UPB member of Romanian Nuclear Energy Association board and expert for the Romanian Nuclear Regulatory Authority. Author or co-author of 17 books and more than 150 papers in journals or conference sessions, he has been working in the energy field related to nuclear and thermal power plant engineering and modeling of nuclear power plant processes and equipment operation.

DANIEL DUPLEAC was born in Bucharest, Romania in 1963. He studied nuclear power plant engineering at UPB and obtained his degrees in 1987 and Doctor of Science Diploma in 1998. He worked for a couple of years for the Cernavoda Nuclear Power Plant and Center of Engineering and Technologies for Nuclear Projects. Since 1990 he teaches at UPB and conducts research in the fields of thermal hydraulic analysis of CANDU reactor and systems, safety analysis of nuclear reactors, modeling of two-phase flow in fuel channel, modeling of CANDU systems.

ADRIAN CONSTANTINESCU was born in Bucharest, Romania and graduated the UPB in 2000. At present he is professor assistant at the UPB where he complete his Ph.D. thesis.

MODELLING OF LAYERED MICROFLUIDIC FLOW IN CONICAL NOZZLE AND DIFFUSER

Manisah Aumeerally and Renate Sitte
School of Information Technology
Gold Coast Campus
Griffith University PMB50
Gold Coast Mail Centre
Queensland 9726, Australia
E-mail: m.aumeerally@griffith.edu.au

KEYWORDS
Micropump, nozzle, diffuser, micro-fluid flow model, electronic circuit modelling

ABSTRACT

The micropump is a microfluidic MEMS device which is used to deliver a precise and minute amount of liquid. Nozzles and diffusers form part of a micropump with no-moving parts. They function as passive valves, changing the direction of flow; hence their flow characteristics are important in the design of the micropump. This paper describes the modeling of fluid flow in conical diffuser and nozzle as an electrical network. It contributes to the physical component of our Virtual Reality-prototyping CAD tool for MEMS, with emphasis on fast calculations for VR representations.

With the underlying fluid flow model based on traditional continuum theory and assuming steady laminar flow, the channel is segmented into layered sections, each having a different flow rate. The flow characteristics of these sections are modeled as a simple electrical circuit. Simulink (Matlab ©) is used to simulate the behaviour of the model. To validate the model we compare its result with experimental results reported in the literature.

INTRODUCTION

The demand for ever smaller, thinner and more compact electronic products lead to a miniaturization revolution in the electronic and related industries. Microelectromechanical system devices such as microsensors, micropumps and micromixers are now commercially available. In biochemical systems, with miniaturization, smaller systems require smaller sample sizes. For example in chemical analysis kits large surface to volume ratio can take advantage of surface chemistry and batch processing, leading to cheaper products. Apart from high throughput, minute amount material requirement and low manufacturing cost, other advantages include low operating and maintenance cost, low power consumption, reduced waste, disposability and reduce risk of contamination, increased precision and accuracy. Some of these devices, known as microfluidic devices such as the micropump, uses fluid flow. These devices are often the components of an integrated microfluidic chip or "lab on a chip", forming part of a miniaturized chemical analysis system (Richter 1997), a micro total analysis system or a micro dosing system (Nguyen 1998) that can have dimensions in the range of 10 μm and are capable of delivering fluid in the nanolitres scale.

With the rapid growth in the demand for such devices, there is a need to develop CAD tools that provide for quick and yet accurate design tool (Sitte2003). Finite Element Modelling (FEM) software such as ANSYS Flotran ™ provides detailed analysis of flow. This often requires hours of work involving the creation of the physical model, meshing, convergence monitoring and several iterations before satisfactory solution is obtained. The objective of this paper is to develop a fast first approximation model for the flow of fluid in a microdiffuser or micronozzle which will be used in an interactive Virtual Reality prototyping CAD tool (Aumeerally 2001). The underlying fluid model is based on the Navier-Stokes equations.

Nozzles and diffusers form part of the valveless diffuser micropump as in (Andersson et al 2000),(Jang et al 2004) and (Jiang et al 1998). These act as passive valves, allowing fluid to flow in one direction only. It should ideally have zero resistance in one direction and infinite resistance in the other direction. Nozzles are channels that have slowly converging walls while diffusers have gradually expanding walls.

As a fluid goes through a nozzle, its flow velocity increases while its static pressure along the axis decreases. The fluid accelerates through the nozzle and the inlet velocity profile distortions become smoothed. Nozzles also reduce turbulence.(Schetz and Fuhs 1999)

Diffusers were designed to increase pressure and reduce kinetic energy. As a fluid enters a diffuser, the flow velocity decreases and the static pressure increases. Diffusers can be conical, pyramidal or planar. The flow in a diffuser is complex and is dependent on several factors such as the diffuser's geometric parameters, inlet Reynolds number, inlet

boundary-layer blockage factor, inlet turbulence, and pulsations. (White 1994)

LAYERED FLOW MODEL

Fluid flow in MEMS is of low Reynolds number (Re), with viscous force being more dominant and inertial force negligible. Consider Poiseuille flow in a circular channel where the velocity distribution is given by:

$$u(x,r) = \frac{R^2 - r^2}{4\mu}\left(-\frac{dp}{dx}\right) \quad (1)$$

where R is the radius of the channel, μ is the viscosity of the fluid and p is the hydrostatic pressure.
The pressure difference between the ends of the channel is given by:

$$p_1 - p_2 = \frac{8\mu}{\pi} Q \int_{x_1}^{x_2} R^{-4}\, dx \quad (2)$$

For flow in a converging-diverging channel where the inertia forces are negligible, equation (3) can be used (Batchelor 1967) where R is the variable radius and is a function of x.

If α is the aperture angle for the diverging section of the channel (region 2 in Figure 1), and is small (<10°), then by substituting

$R = R_o + \alpha x$ where R_o is the radius at the diffuser's throat into equation (1), we obtain

$$\Delta P_2 = \frac{8\mu QL}{3\pi R_o^4}\left(\frac{3 + 3\lambda + 3\lambda^2}{(1+\lambda)^3}\right) \quad (3)$$

where ΔP_2 is the pressure difference between the end of the channel and the throat, L is the length of the channel and

$\lambda = \alpha L / R_o$.

For region 1 the pressure difference is given by (Heschel et al 1997) equation (4):

$$\Delta P_1 = \frac{8\mu Q}{\pi}\left(-\frac{1}{R_i^3}\right)\left(\frac{6\beta^4 + 10\beta^2 - 1}{6\beta^2(\beta^2-1)^3} + \frac{(4\beta^4+1)\tanh^{-1}\left(\frac{1+\beta}{\sqrt{1-\beta^2}}\right)}{(1-\beta^2)^{3.5}}\right) \quad (4)$$

where R_i is the radius at the inlet and

$$\beta = \frac{R_o}{R_i} + 1$$

The total pressure difference across the diffuser with the curved inlet is the sum of the pressure difference in region 1 and region 2 ie $\Delta P = \Delta P_1 + \Delta P_2$
If the resistance $\Re = \Delta P / Q$, then:

$$\Re = \frac{8\mu L}{\pi}\left(\frac{3 + 3\lambda + 3\lambda^2}{3R_o^4(1+\lambda)^3} - \frac{1}{R_i^3}\left(\frac{6\beta^4 + 10\beta^2 - 1}{6\beta^2(\beta^2-1)^3} + \frac{(4\beta^4+1)\tanh^{-1}\left(\frac{1+\beta}{\sqrt{1-\beta^2}}\right)}{(1-\beta^2)^{3.5}}\right)\right) \quad (5)$$

For the inner and middle fluid layers in the diffuser, we assume that the flow is almost that of a circular channel and the resistance for each sections can be determined by (Aumeerally & Sitte 2003):

$$\Re = \frac{8\mu L}{\pi\left[2a^2(r_2^2 - r_1^2) - (r_2^4 - r_1^4)\right]} \quad (6)$$

where a is the radius of the channel.
The resistance of the outer layer is the difference between the total resistance using equation (5) and the sum of the resistances of the inner and middle layers using equation (6).

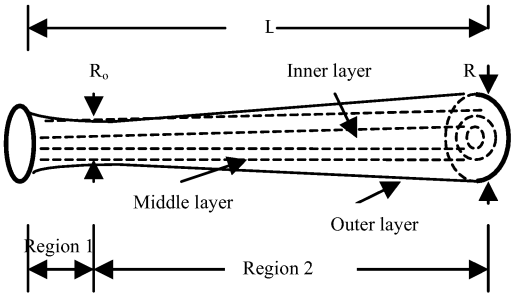


Figure 1 The two regions and the three layers of fluid in a conical diffuser

MODELLING AS AN ELECTRICAL CIRCUIT

As in the layered flow model of a circular channel (Aumeerally & Sitte 2003), volume flow is represented by the current, pressure difference is represented by the voltage, viscous forces by the resistance, and the mass of the fluid is represented by the inductance. Assuming that the fluid flow is laminar, steady and that no flow separation occurs, the flowrate in the diffuser will be the same as for the nozzle, which is just the diffuser with fluid flowing in the other direction. We can represent the flow of the layers for the nozzle (or diffuser) by resistances in series and in parallel as shown in figure 2. Each branch of the electric circuit represents the flow for each fluid layer and is the sum of the resistances in region 1 and region 2 of the diffuser. The current flow for each layer is calculated by using current division rule:

$$i_n = \frac{1/\Re_n}{\sum_{n=1}^{N} 1/\Re_n} i_s \qquad (7)$$

where n is the number of layers, i_s is the total current and $\Re_n$ the sum of the resistances for each branch.

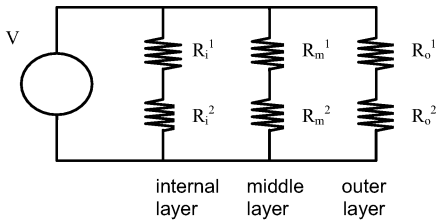


Figure 2 Electric circuit representation for the fluid layers in sections of the diffuser

The electrical model of figure 3 is implemented in Simulink in the same way as for the layered flow in a circular channel (Aumeerally and Sitte 2003).

The values of the flowrates obtained for the microchannel and nozzle (or diffuser) are used as inputs into our VR module for an animated display. The resistances obtained for the microchannel are used as inputs for the diffuser into the next Simulink model of figure 4, together with the resistances of flow in the nozzle (or diffuser) direction.

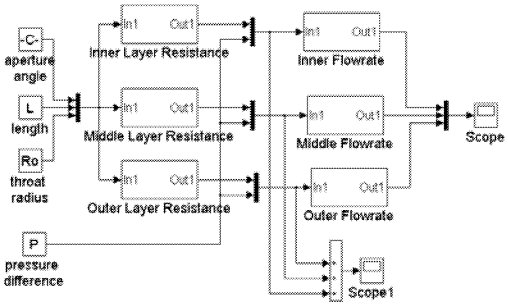


Figure 3 Simulink model for the flowrates and resistances for the three fluid layers in the diffuser

From the model of figure 4, the efficiency of the nozzle/diffuser pair used as valves can be determined from the ratio of the conductance for the flow in the nozzle direction to that in the diffuser direction. The efficiency is between 0 and 1. (Olsson et al. 1996) (Gerlach 1998) (Singhal et al. 2004)

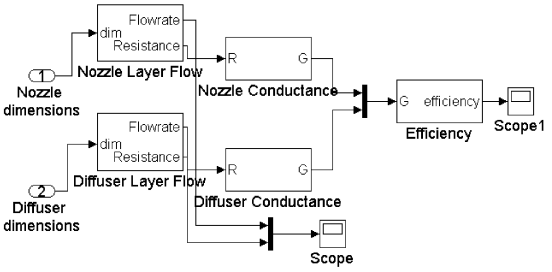


Figure 4 Simulink model for the efficiency of nozzle/diffuser valves

EXPERIMENT AND RESULTS

As an example, a conical diffuser of length 1200 μm, aperture angle 6.3° and throat radius 10 μm is used (Heschel et al 1997). The flowrates for a range of pressure difference are calculated using equation (3) and (4) and these are shown in figure 5. The flowrates calculated using the electrical model of equations (5)-(7) are also shown in the same figure. As can be seen, the electrical model gives exactly the same values as the analytical ones. However when compared with experimental data, both the analytical model traditionally used, as well as the electrical model give values that deviate increasingly more as the pressure difference goes above 100 mbar.

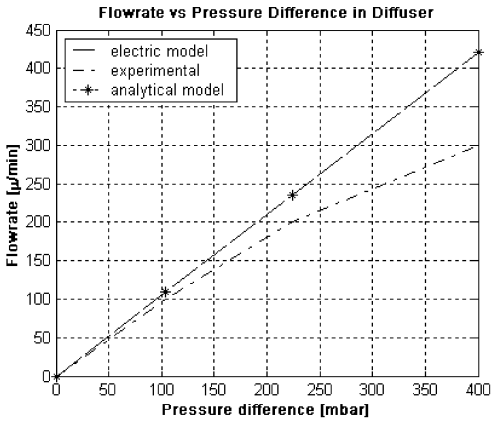


Figure 5 Relationship between flowrates and pressure difference using experimental data, electric and analytic model.

VALIDATION AND DISCUSSION

A comparison is made between values of the flowrates and are shown in a chart in figure 6. The percentage differences between the experimental data and the values of the electrical and analytical model are shown in Table 1. As before, there is no difference between the electrical model values and the analytical values.

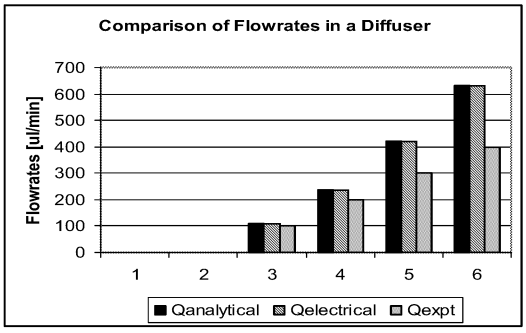


Figure 6 Comparison of flowrates from the electrical and analytical model and experimental data

The percentage error increases with the flowrates. The resistance is a linear function of pressure and flowrate as can be seen from equation (5). From the experimental values it appears that the resistance is not a linear function. A quadratic term needs to be added to the equation to give a non-linear function. Energy loss has to be accounted for.

Qanalytical	Qelectrical	Qexpt	Error %
0	0	0	
109.5	109.5	100	8.7
235.8	235.8	200	15.2
421.2	421.2	300	28.8
631.7	631.7	400	36.7

Table 1 Percentage errors of flowrates from electrical model to experimental values

In using the electrical model, the assumptions are that the flow is laminar, there is no turbulence and no flow separation. Laminar flow in a diffuser occurs only for very small diverging angles (less than 10°) and small exit area to throat area ratio. In order to achieve high efficiency, the pressure loss has to be minimum, and this occurs in what is known as the transitory stall region of the stability map of a diffuser. In this region, the flow pattern is highly unsteady, with large pressure fluctuations, reversed flow and flow separation. (Stemme and Stemme 1993)

CONCLUSION

In this paper we presented an electrical model for the fast calculation of the flow rates of steady laminar fluid flow in a diffuser. The diffuser is divided up into 2 regions, at the inlet and the diverging sections. The fluid is divided up into 3 layers with the central 2 layers parallel to the axis. An electrical network model was derived and implemented in Matlab/Simulink. These results were compared with experimental data from the literature. At low pressure difference, the percentage errors in flowrates are small (less than 10%) but this increases at higher pressure.

The electrical network consisting of series and parallel resistances provides a suitable model for the flow rate of a liquid in steady laminar flow in a conical diffuser at low pressure differentials. Future work aims at incorporating energy dissipation to the model so that it can be applied to a wider range of pressure differentials.

REFERENCES

Richter,M.; A.Prak; J.Naundorf.; M.Eberl; P.Woias and A.Steckenborn. 1997. "A Chemical Microanalysis System as a Microfluid System Demonstrator." *Proceeding of IEEE 1997 International Conference on Solid-State Sensors and Actuators,* Chicago, 303-306

Nguyen,N.T.; S.Schubert, S.Richter and W.Dötzel. 1998. "Hybrid-assembled MicroDosing System Using Silicon-based Micropump/valve and Mass Flow Sensor." *Sensors and Actuators A,* vol. 69, 85-91

M. Aumeerally, R. Sitte, "Design and Modelling of Microactuators in MEMS Virtual Reality Prototyping CAD tool", *Electronics and Structures for MEMS II*, Neil Bergman, Editor, Proceedings of SPIE Vol 4591, pp 163-171, 2001

Andersson, H.; W. V. D Wijngaart; P.Nilsson; K.Enoksson and G.Stemme. 2001. "A Valveless Diffuser Micropump For Microfluidic Analytical Systems.' *Sensors and Actuators A,* vol. 72, 259-265

Jang,W.I.; C.A.Chooi; C.H.Jun; Y.T.Kim and M.Esashi. 2004. "Surface Micromachined Thermally Driven Micropump." *Sensors and Actuators A*, vol. 115, 151-158

Jiang,X.N.; Z.Y.Zhou; X.Y.Huang; Y.Li; Y.Yang and C.Y.Liu. 1998. "Micronozzle/diffuser and its Application in Micro Valveless Pumps." *Sensors and Actuators A*, vol. 70, 81-87

Schetz, J.A and A.E.Fuhs, editors. 1999. "Fundamentals of Fluid Mechanics." John Wiley & sons, NY.

White, F. 1994. "Fluid Mechanics." Mc-Graw Hill, USA.

Batchelor, G.K. 1967. "An Introduction To Fluid Dynamics." Cambridge University Press, UK.

Heschel,M.; M.Müllenborn and S.Bouwstra. 1997. "Fabrication and Characterization of Truly 3D Diffuser/Nozzle Microstructures in Silicon." *Journal of Microelectromechanical System, vol. 6, no. 1,* March 1997

Aumeerally, M. and R. Sitte. 2003. "Simulation and Modelling of Layered Fluid Flow in a Rectangular Microchannel" Middle East Symposium on Simulation and Modelling MESM 2004, pp 81 - 85

Aumeerally, M. and R. Sitte. 2003. "Modelling of a Layered Flow in a Circular Microchannel." *Proceedings of ESMC 2003,* Naples pp 458-462.

Olsson,A.; G. Stemme and E. Stemme. 1996. "Diffuser-element Design Investigation for Valveless Pumps." *Sensors and Actuators A,* vol.57, 137-143.

Gerlach, T. 1998. "Microdiffusers as a Dynamic Passive Valves for Micropump Applications." *Sensors and Actuators A,* vol. 69, 181-191.

Singhal, V.; S.V. Garimella and J.Y. Murthy. 2004. "Low Reynolds Number Flow Through Nozzle-Diffuser Elements in Valveless Micropumps." *Sensors and Actuators B,* vol. 9, 152-161.

Stemme, E and G.Stemme. 1993. "A Valveless diffuser/nozzle-based Fluid Pump." *Sensors and Actuators A,* vol. 39, 159-167.

LATE PAPERS

STREAMS OF EVENTS AND PERFORMANCE OF QUEUING SYSTEMS: THE BASIC ANATOMY OF ARRIVAL/DEPARTURE PROCESSES, WHEN THE FOCUS IS SET ON AUTOCORRELATION

Erland Hejn Nielsen
Department of Accounting, Finance and Logistics
Aarhus School of Business
Fuglesangs Allé 4
DK-8210 Aarhus V, Denmark
E-mail: ehn@asb.dk

KEYWORDS

Autocorrelation, queuing systems, TES method, ARTA method, Descriptive/Selective sampling, Simulation, Job/flow-shop, performance, control.

ABSTRACT

Judging from the vast number of articles in the field of queuing simulation, that assumes i.i.d. in one or more of the stochastic processes used to model the situation at hand, often without much validation, it seems that sequence independence must be a very basic property of many real life situation or at least a very sound approximation.

However, on the other hand, most actual decision making is based upon information taken from the past - where else! In fact the only real alternative that comes into my mind is to let a pair of dices fully and completely rule behaviour, but I wonder if such a decision setup is that widespread in consequent use anywhere. So, how come that sequence independence is so relatively popular in describing real system processes?

I can only think of three possible explanations to this dilemma - (1) either sequence dependence is present, but is mostly not of a very significant nature or (2) aggregate system behaviour is in general very different from just the summing-up (even for finite sets of micro-behavioural patterns) and/or (3) it is simply a wrong assumption that in many cases is chosen by mere convention or plain convenience.

It is evident that before choosing some arrival processes for some simulation study a thorough preliminary analysis has to be undertaken in order to uncover the basic time series nature of the interacting processes. Flexible methods for generating streams of autocorrelated variates of any desired distributional type, such as the ARTA method or some autocorrelation extended descriptive sampling method, can then easily be applied. The results from the Livny, Melamed and Tsiolis (1992) study as well as the results from this work both indicates that system performance measures as for instance average waiting time or average time in system are significantly influenced by the taken i.i.d. versus the autocorrelations assumptions. Plus/minus 35% in performance, but most likely a worsening, is easily observed, when comparing even moderate (probably more realistic) autocorrelation assumptions with the traditionally and commonly used i.i.d. assumptions.

INTRODUCTION

Judging from the vast number of articles in the field of queuing simulation, that assumes i.i.d. in one or more of the stochastic processes used to model the situation at hand, often without much validation, it seems that sequence independence must be a very basic property of many real life situation or at least a very sound approximation.

However, on the other hand, most actual decision making is based upon information taken from the past - where else! In fact the only real alternative that comes into my mind is to let a pair of dices fully and completely rule behaviour, but I wonder if such a decision setup is that widespread in consequent use anywhere. So, how come that sequence independence is so relatively popular in describing real system processes?

I can only think of three possible explanations to this dilemma - (1) either sequence dependence is present, but is mostly not of a very significant nature or (2) aggregate system behaviour is in general very different from just the summing-up (even for finite sets of micro-behavioural patterns) and/or (3) it is simply a wrong assumption that in many cases is chosen by mere convention or plain convenience.

To put this discussion further into perspective it can be noted, that methods for introducing autocorrelation into a simulation study are in fact numerous. The ARTA-method suggested by Cario and Nelson (1997), where ARTA denotes "Auto-Regressive-To-Anything", and the TES-method developed by Jagerman and Melamed (1992,a & b), where TES denotes "Transform Expand Sample", are nice representatives of the various methods for generating uniform variates, that incorporates autocorrelation. Even finite sets of variates methods, as for instance "selective" sampling originally developed by Brenner (1963), or "descriptive" sampling as originally described by Saliby (1989), may at least in principle also be used to create autocorrelation patterns of almost any imaginable nature, if only it is possible to devise a relevant "scrambling" procedure with a resulting desired sequence dependence.

There are, in my opinion, several good reasons for the need to look closer into the phenomenon of process autocorrelation in general and in relation to queuing and job/flow shop systems in particular. Autocorrelation is in my view, and I find it well supported by practical experience, a much more predominant phenomenon in socio-economic systems, than is the case of independence. And results presented by Livny, Melamed and Tsiolis (1992) corroborates this view that it is immensely important to take proper account

of any potential autocorrelation present, by showing that autocorrelation mostly has a profound negative effect on the functioning of a simple queuing system.

One could also think of another situation where we quite deliberately chooses to introduce autocorrelation into some event stream, namely whenever the "Shortest Processing Time" rule, in short the SPT-rule, is applied for prioritizing jobs in a queue. This is in contrast a situation where autocorrelation is systematically introduced and utilized in order to explicitly improve the overall system performance.

So autocorrelation has seemingly both the potential to introduce positive as well as negative effects into the systems functioning and so it obviously has to be of interest to know more about the basic anatomy of event streams with respect to autocorrelation and also how some given autocorrelation phenomenon can be expected to propagate its way through for example a job/flow-shop system.

Let's therefore approach the discussion of queuing systems and the relevance of autocorrelation by first clarifying, that there can of course be good reason sometimes to assume independence in event processes, typically arrival processes. From the Palm-Khintchine Theorem (1969) we know that under fairly mild conditions on the individual (n'th) arrival source (s_n), that the "superposition process" that results from the mixing of all the (s_n) processes with $n \to \infty$ is known to converge towards a Poisson process. This is popularly speaking sort of a "Central Limit Theorem" for arrival processes, that kicks in and govern the nature of mixtures of an infinity of independent sub-processes.

However, when the number of processes that is mixing is limited, or as it often will be the case in reality, consists of only a few processes say 1 or 2, sequence independence of event inter-arrival times of the mix-process is not guaranteed by any means. In fact, if the individual process components in the mix process show sign of autocorrelation, then certainly also does the mix-process to some extent.

The present work will take its offset in the results presented by Livny, Melamed and Tsiolis (1992) and start asking the question as to what extent these findings simply can be ascribed to the level of the 1st order autocorrelation or if they are dependent on the characteristics of the whole TES-setup? It will be tested by recomputing some of the Livny, Melamed and Tsiolis results with an alternative, more flexible method for generating autocorrelated variates with comparable 1st order autocorrelations, but otherwise significantly different ACF/PACF profile, where by the way ACF denotes the "AutoCorrelation-Function" and PACF denotes the "Partial AutoCorrelation-Function". Livny, Melamed and Tsiolis (1992) themselves also investigates one alternative method for generating autocorrelated uniform variates, called the Minification method, however, though the Minification method is different from the TES method with respect to the ACF/PACF profiles, it is just as inflexible as the TES method, with respect to the shaping of the full autocorrelation profile. The more flexible "TES-deviating" setups, that will be considered in this work will be discussed in two variants, one that obeys a certain distributional shape asymptotic (ARTA) and one that obeys a certain distributional shape relative to a given finite sample exact distributional form (Extended Descriptive Sampling with Autocorrelation). The last setup effectively separates, at least in principle, the autocorrelation effect onto the performance in its most pure form.

Having, hopefully, by now established whether just a first order approach to autocorrelation in event streams essentially is of importance or a more full approach is called for, this work will continue by taking an analytical offset in the autocorrelation characteristics of some output stream of events from a simple M/M/1 queuing system operated by some specific queuing discipline as for instance the SPT-rule. The question that comes naturally in this situation is whether TES, ARTA etc. are methods that can be fitted well enough to approximate/simulate the input to succeeding systems, that otherwise typically would take the output from a few simple M/M/1-like queuing systems, operated, of course, by some given queuing discipline, as input?

Autocorrelation in Event Streams

Whether an event process is autocorrelated or not can seldom be judged simply by looking at its graphical appearance. Let for example x be i.i.d. exponential distributed and let $X_n = X_{n-1} + x_n$ be the n'th x-event time of occurrence ($X_0 = 0$). Let further y be not independent but still i.d. exponential distributed and let $Y_n = Y_{n-1} + y_n$ be the n'th y-event time of occurrence ($Y_0 = 0$). Plotting the event streams for both these processes will look like the following, where each "tic" denoting an event

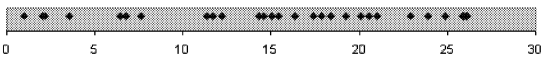


Figure 1: Occurrence of "x"-events

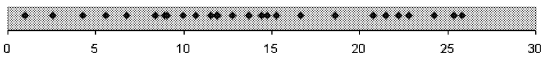


Figure 2: Occurrence of "y"-events

Clearly, the sample is too small to make any judgement at all, however, if on the other hand the sample was larger the eye would not be able to see anything, but a mess of "tics".

So, there is called for a more elaborate approach in order for example to uncover a possible first order autocorrelation effect. A plot of x against its lagged value might give some insight.

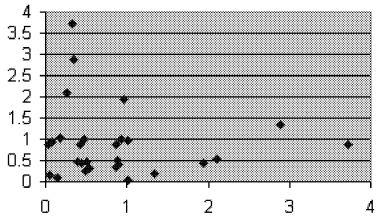


Figure 3: x against lag(x)

However, despite the pictures does not give any conclusive information, due to a still too small a sample size, they non the less appear to be of a distinctive different nature. Further performing a plot of the autocorrelation as well as the

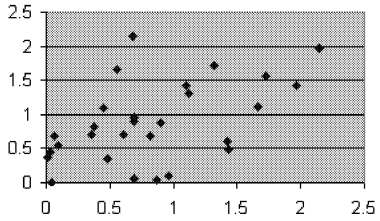


Figure 4: y against lag(y)

partial autocorrelation functions for this very scarce sample size does, as should be expected, not add significantly to the findings above, remembering that 5% of the observations are to be expected to be significant by mere chance.

Finally, please observe that in both plots (Figure 5 & 6), as will also be the case whenever this type of ACF/PACF graphical representation is applied elsewhere in this paper, the "Series 1"-label denotes the ACF profile, the "Series 2"-label denotes the PACF profile and the "Series 3 & 4" denotes the 95% confidence limits for both the ACF and the PACF profiles.

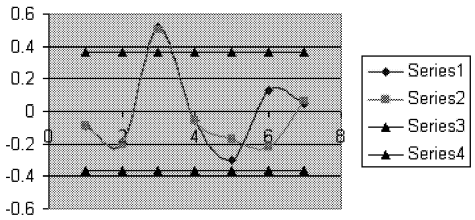


Figure 5: ACF and PACF for x

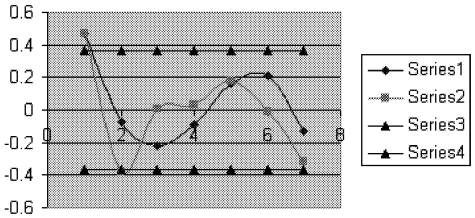


Figure 6: ACF and PACF for y

As a contrast to the above (lack of) findings, if we repeat the plotting of the autocorrelation as well as the partial autocorrelation functions based now on 20000 observations, we get a whole lot of a very different story told. The ACF and PACF plots in figure 7 & 8 tells us that the x-variate is clearly i.i.d. whereas the y-variate is most likely an AR(2)-process with $\rho_1 \approx 0.4$ and $\rho_2 \approx -0.2$. By the way, if necessary, the parameters of the underlying AR(2)-process can easily be computed from the ρ values by exercising the socalled Yule-Walker equations (see for a full description).

So, let us now turn our attention towards the output processes emanating from queuing systems, which typically is the input to succeeding stages in a production flow. Let us begin by taking a closer look in terms of autocorrelations at the stream of events flowing from a simple M/M/1 queuing system governed by different queue priority rules such as for instance SPT and LPT, where LPT denotes "Largest

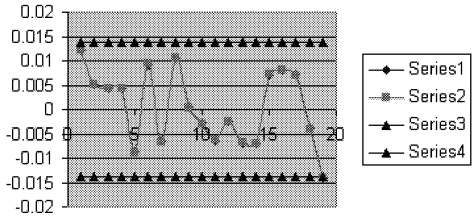


Figure 7: ACF and PACF for x

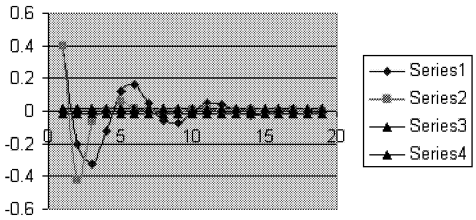


Figure 8: ACF and PACF for y

Processing Time". The different setups will be considered for different levels of traffic intensity - say 0.25, 0.75 and 0.95.

The following six M/M/1 queuing system setups, which are "x-rayed" below (figure 9 through 14) for their autocorrelation properties, are all simulated for a pass-through of 20000 units, customers or items of goods if you like.

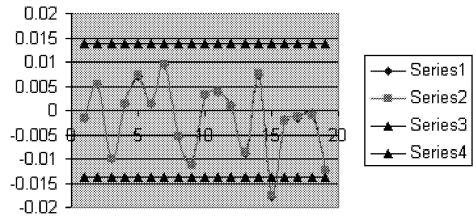


Figure 9: ACF and PACF for M/M/1/SPT output - traffic intensity=0.25

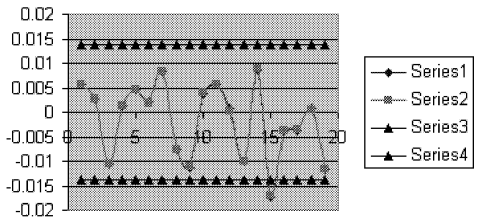


Figure 10: ACF and PACF for M/M/1/LPT output - traffic intensity=0.25

As should be expected the filtering effect of a given queue priority setup only kicks in for relative high levels of utilisation. For traffic intensities of 0.75 and 0.95 we see a pronounced autocorrelation generating effect by both the SPT and LPT rule. Negative autocorrelations for the SPT-rule, and positive ones for the LPT-rule. It can also be noticed that in the SPT case the first order autocorrelation coefficient is absolutely and almost solely dominant, whereas in the LPT case a much more slowly decreasing full set of autocorrelation coefficients is the case.

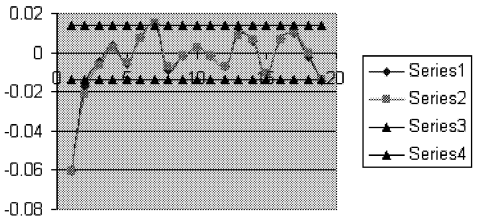


Figure 11: ACF and PACF for M/M/1/SPT output - traffic intensity=0.75

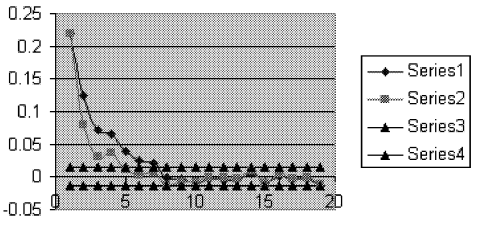


Figure 12: ACF and PACF for M/M/1/LPT output - traffic intensity=0.75

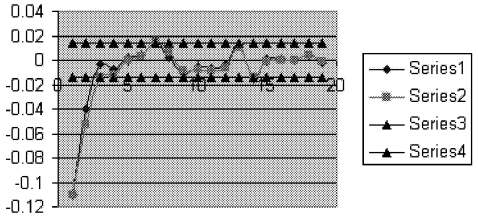


Figure 13: ACF and PACF for M/M/1/SPT output - traffic intensity=0.95

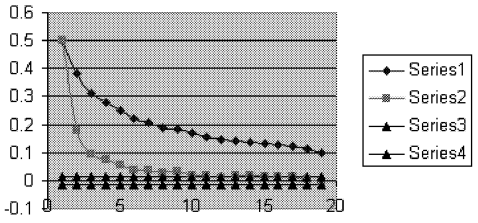


Figure 14: ACF and PACF for M/M/1/LPT output - traffic intensity=0.95

The simulation results reported by Livny, Melamed and Tsiolis (1992) in their paper with the title "The Impact of Autocorrelation on Queuing Systems" mainly tells us that it is autocorrelation in the arrival process that hurts the most and definitely more than autocorrelation in the service process, at least for M/M/1/FIFO systems. Both possitive as well as negative autocorrelation patterns has according to their findings a strong deteriorating effect on typical performance measures, such as for example the average waiting times, whenever the system utilization level is moderate to high.

From the autocorrelation plots (Figure 11 & 13) for the M/M/1/SPT queuing system it seems clear that the first order autocorrelation plays the dominant role which implies that it is important to be able to model autocorrelation in arrival processes by the magnitude of this first order autocorrelation relationship, but is this sufficient? Or is there a reasonable need for the ability to be able to model higher order autocorrelation properties? The TES method, the one that is used by Livny, Melamed and Tsiolis (1992), for generating uniformly distributed variates is computational extremely effective in a simulation context, but allows only for the specification of the first order autocorrelation coefficient by which the whole ACF/PACF structure is determined. It would of course, primarily due to the computational aspect, be nice if TES-type autocorrelation processes could be viewed as sound and generally applicable approximations, a kind of first order approximation, to any type of autocorrelation present in a given analysis. This is unfortunately not the case! We consequently have to look broader. The ARTA method for generating autocorrelation properties in event processes is much more flexible in that it allows in principle the specification of the full spectrum of autocorrelation coefficients, however confined to the class of covariance-stationary autoregressive time series models, where the Yule-Walker equations constitutes a solid link between the autocorrelation coefficients and the specification of the data generating AR-process. The ARTA method is in principle as computational efficient and easy to use as the TES method.

The following section will deal with and dig a little bit deeper into the differences between the TES- and the ARTA methods as judged by their impact on typical performance measures relating to the M/M/1/FIFO queuing system.

TES versus ARTA

The TES method is a very efficient method for generating autocorrelated uniform variates given by the following transformation scheme that can be built into any good uniform random number generator.

$$\begin{aligned} U_0^+ &= Z_0 \\ U_i^+ &= \langle U_{i-1}^+ + L + (R-L)\cdot Z_i \rangle \\ U_i^- &= U_i^+ \quad \text{if } i \text{ is even} \\ U_i^- &= 1 - U_i^+ \quad \text{if } i \text{ is odd} \\ \text{for } i &= 1, \ldots, N \end{aligned} \tag{1}$$

where the Z_i are i.i.d. uniform random variates and $-0.5 < L < R \leq 0.5$ are the pair of parameters that parameterise the TES method. The notation $\langle x \rangle$ denotes modulo-1 arithmetic. The two variates U_i^+ and U_i^- are respectively the positive and negative autocorrelated TES variates. Some example of TES-ACF/PACF paths are illustrated in the figures 15 & 16.

The ARTA method (AutoRegressive-To-Anything) is a method based on the standard covariance stationary autoregressive time series setup

$$Y_i = const + \alpha_1 \cdot Y_{i-1} + \ldots + \alpha_n \cdot Y_{i-n} + \epsilon_i \tag{2}$$

where ϵ_i are normally distributed. Due to the Yule-Walker equations a well defined relation between the autocorrelation coefficients and the process parameters exists for this type of time series processes, which makes them very well suited for generating variates of any type. The Y variate is normally distributed and has consequently a known distribution function that takes Y into the uniform domain, and from there it is standard to transform to any other type of distribution. To make sure that the desired autocorrelations prevail in the final distribution ARTA makes the necessary corrections to the "Yule-Walker" given AR parameters. Some example of ARTA-ACF/PACF paths are illustrated in the figures 17 through 20.

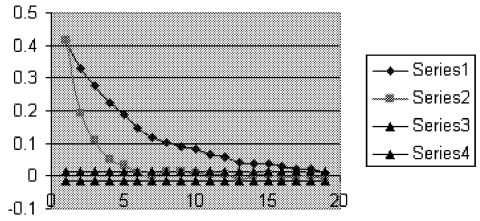


Figure 15: ACF and PACF for TES($\rho_1 = 0.4$) generated arrival stream - Exponential distributed with mean=1

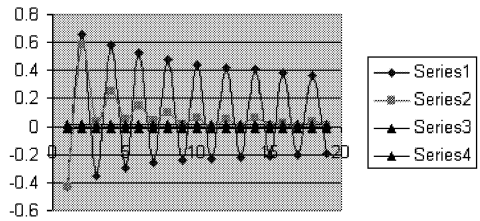


Figure 16: ACF and PACF for TES($\rho_1 = -0.4$) generated arrival stream - Exponential distributed with mean=1

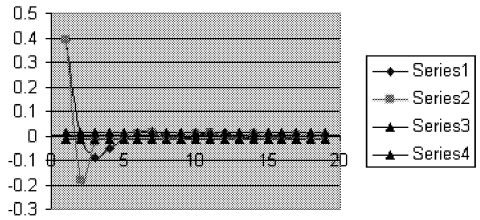


Figure 17: ACF and PACF for ARTA(a) generated arrival stream - Exponential distributed with mean=1, (see table 1 for further specifications of the process-parameters (the head-column).)

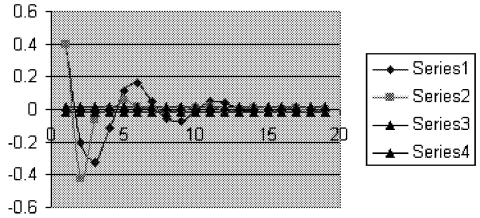


Figure 18: ACF and PACF for ARTA(b) generated arrival stream - Exponential distributed with mean=1, (see table 1 for further specifications of the process-parameters (the head-column).)

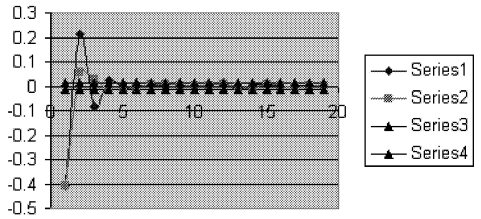


Figure 19: ACF and PACF for ARTA(c) generated arrival stream - Exponential distributed with mean=1, (see table 2 for further specifications of the process-parameters (the head-column).)

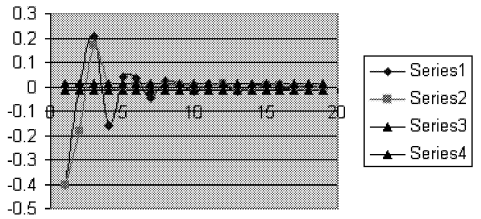


Figure 20: ACF and PACF for ARTA(d) generated arrival stream - Exponential distributed with mean=1, (see table 2 for further specifications of the process-parameters (the head-column).)

The following M/M/1/FIFO system experiments will be conducted with only autocorrelation present in the arrival stream and will be based on the same patterns as described in the figures 15 through 20. The service process will be standard i.i.d. exponential distributed.

The findings reported in table 1 & 2 (below) by and large tells us that positive first order autocorrelations tends to deteriorate performance compared to the i.i.d. base case, whereas negative first order autocorrelations in fact has the potential to improve performance compared to the i.i.d. base case, except for such extreme ACF-patterns, which are implied by the TES-method, that on the contrary causes a very dramatic deterioration in performance. This is strongly in accordance with the results in the Livny, Melamed and Tsiolis (1992) paper.

Traffic Intencity	0.25	0.75	0.95
I.I.D.	0.33 (0.33//0.33)	2.98 (2.92//3.05)	18.80 (17.18//20.42)
TES, $\rho_1 = 0.4$	0.46 (0.45//0.46)	8.45 (8.25//8.65)	60.24 (50.29//70.19)
ARTA(a), $\rho_1 = 0.4$, $\rho_2 = -0.00001$	0.36 (0.36//0.36)	3.60 (3.54//3.66)	22.08 (19.90//24.26)
ARTA(b), $\rho_1 = 0.4$, $\rho_2 = -0.2$	0.35 (0.35//0.35)	2.99 (2.95//3.04)	17.55 (15.90//19.19)

Table 1: Estimated M/M/1/FIFO Average System Time / Flow = 100000 units / Transient Period = 10000 / Replications = 10

Traffic Intencity	0.25	0.75	0.95
I.I.D.	0.33 (0.33//0.33)	2.98 (2.92//3.05)	18.80 (17.18//20.42)
TES, $\rho_1 = -0.4$	0.31 (0.31//0.31)	5.72 (5.38//6.07)	141.32 (104.47//178.17)
ARTA(c), $\rho_1 = -0.4$, $\rho_2 = 0.2$	0.31 (0.31//0.31)	2.35 (2.32//2.39)	13.79 (12.59//14.98)
ARTA(d), $\rho_1 = -0.4$, $\rho_2 = 0.00001$	0.31 (0.31//0.31)	2.25 (2.22//2.29)	13.10 (12.05//14.16)

Table 2: Estimated M/M/1/FIFO Average System Time / Flow = 100000 units / Transient Period = 10000 / Replications = 10

It is also worth noting that the observed performance improvement caused in the cases ARTA(c) and ARTA(d) very well can be and probably has to be attributed to significant higher order (> 2) autocorrelation effects, that is in other words, the full ACF/PACF profile! The applied ARTA specifications are in this study only specified up to the second order and as can be seen in the ARTA(d) case (see table 2), though $\rho_2 \approx 0$, figure 20 tells us that $\rho_3 \approx 0.2$ and $\rho_4 \approx -0.15$! It clearly would be interesting to get a more complete and systematic understanding of the full ACF/PACF patterns and their systematic relation and influence on system performance in general.

Descriptive sampling with Autocorrelation

Descriptive or selective sampling is mostly known as a procedure for scrambling N numbers z_i given by $\frac{1}{2}\left(\frac{2 \cdot i - 1}{N}\right)$ where $i = 1, \ldots, N$, independently by picking numbers in a random fashion based on some well behaved congruent mechanism, which then results in a finite set of independent (truly)uniformly distributed variates u_i where $i = 1, \ldots, N$.

Descriptive sampling is a method that is not accepted by all simulation analysts as being a sound procedure - but I think it offers a method of last resort, in cases where the ARTA method is not able to suggest any generating method at all, given some observed set of data. The autocorrelated (extended) descriptive sampling method can be made working on virtual any set of observed data, no strings attached, which I think is quite a nice property.

A scrambling procedure, whereby some desired autocorrelation property can be incorporated into exactly N numbers u_i is, however, not a stright foreward matter. Non the less, one scheme for scrambling a finite set of figures z_i given by $\frac{1}{2}\left(\frac{2 \cdot i - 1}{N}\right)$ where $i = 1, \ldots, N$ could be as follows:

- Generate or observe N successive values and perform a rescale down to the 0-1-interval, resulting in $y_1, y_2, \ldots, y_N$.
- Now consider a nonlinear mathematical programming problem in $N \times N$ binary variables $\gamma_{i,j}$ given as follows

$$\text{MIN Z} \quad = \quad \sum_{i=1}^{N} \Phi_i^2$$

Subject to

$$\Phi_i \quad = \quad y_i - \sum_{j=1}^{N} \gamma_{i,j} \cdot z_j \quad \text{for } i = 1, \ldots, N$$

$$\sum_{j=1}^{N} \gamma_{i,j} \quad = \quad 1 \quad \text{for } i = 1, \ldots, N \qquad (3)$$

$$\sum_{i=1}^{N} \gamma_{i,j} \quad = \quad 1 \quad \text{for } j = 1, \ldots, N$$

$$\text{all } \gamma_{i,j} \text{ are binary}$$

In essence the solution to this optimization problem will produce the best-fit permutation of the z data in relation to the generated y sample. Solution values $\gamma^*_{i,j} = 1$ denotes the placement of z_j at sequence index i, that is $u_i = \sum_{j=1}^{N} \gamma^*_{i,j} \cdot z_j$ where $i = 1, \ldots, N$.

Unfortunately, the above formulated optimization problem is quite cumbersome to solve exactly in practice. However, an approximate solution method, based on a repeatedly pair-wise interchange of z values in order to step-wise minimize $\sum_{i=1}^{N} \Phi_i^2$, has "proven" in my experience to "copy" enough of the autocorrelation characteristics from the y sample, in a manageable number of iterations, to be a satisfactory procedure. But still this partial interchange method is not a computational very efficient method and the computational effort increases by N^2. The generation of 500 variates is still manageable, but beyond this number it becomes rapidly quite impractically.

The results from using this method is showing the same general tendency as the results from the TES and the ARTA method, when these specific characteristics are "copied", even on very small data sets, but as the computational efficiency is very low, I will not continue to present any more detailed analysis in this study based on this extended descriptive sampling method.

Concluding Remarks

In the Livny, Melamed and Tsiolis (1992) paper the conclusion is amongst others that TES should be utilized whenever the analyst is in need for a conservative benchmark on the systems performance. This makes sense because by comparing the results in this work especially figure 12 & 14 with figure 15 it can be seen that TES operated with a significant positive first order autocorrelation effect is much the same as if the arrival generating process was another queue-sub-system ruled by the not very efficient "units with the largest processing time come first" or in short the LPT rule.

Now taking a summarising look at the autocorrelation patterns generated by a queue-sub-system ruled by the SPT-rule it seems that much resemblance can be found when comparing the figures 11 & 13 with the figures 19 & 20. The performance results for these instants are also quite reassuring, given common experience, and in addition they give a slightly deeper understanding of the reasons for the popularity of the SPT-rule in terms of autocorrelation patterns.

First order negative autocorrelation is obviously beneficial, however, from the figures 16 & 20 and their corresponding table-results it is quite clear that full autocorrelation pattern is of vital importance for the overall systems performance outcome.

It is evident that before choosing some arrival processes for some simulation study a thorough preliminary analysis in order to uncover the basic time series nature of the interacting processes must be undertaken. Having done so, flexible

methods for generating streams of autocorrelated variates of any desired distributional type such as the ARTA method or some autocorrelation extended descriptive sampling method can more or less easily be applied, and as the results from the Livny, Melamed and Tsiolis (1992) study as well as the results from this work indicates, the system performance measures are heavily influenced by the i.i.d. versus the autocorrelations assumptions done. Plus/minus 35% in performance is easily observed even when comparing moderate and probably more realistic autocorrelation assumptions with the traditionally and commonly used i.i.d. assumptions.

REFERENCES

Brenner, M.E., 1963. "Selective sampling - a technique for reducing sample size in simulations of decision making problems." *Journal of Industrial Engineering*, 14, 291-296.

Cario, M.C., and B.L. Nelson, 1996. "Autoregressive to anything: Time-series input processes for simulation." *Operations Research Letters*, 19, 51-58).

Cario, M.C., and B.L. Nelson, 1997. "Numerical Methods for Fitting and Simulating Autoregressive-To-Anything processes." *(Working Paper????).*

Box, G.E.P., and G.M. Jenkins, 1976. "Time Series Analysis: Forecasting and Control." *Holden-Day Inc.*.

Jagerman, D.L., and Melamed, B., 1992. "The transition and autocorrelation structure of TES Processes Part I: general theory." *Stochastic Models* 8(2):193-219.

Jagerman, D.L., and Melamed, B., 1992. "The transition and autocorrelation structure of TES Processes Part II: special cases." *Stochastic Models* 8(3):499-527.

Kelton, W.D., Sadowski, R.P., Sadowski, D.A., 2002. "Simulation with ARENA" (2nd ed.) *McGraw-Hill International Edition.*

Khintchine, A.Y., 1969. "Mathematical Methods in the Theory of Queueing" (2nd ed.) *Griffin's Statistical Monographs & Courses, London.* First published in English 1960 where it was translated from the Russian text first published under the title *"Matematicheskie metody teorii massovogo obsluzhivaniya"* to be found in Vol. 49 (1955) of the series Trudy Matematicheskova Instituta Steklov, Akad. Nauk, U.S.S.R..

Law, A.M., and Kelton, W.D., 2000. "Simulation Modeling and Analysis" (3rd ed.) *McGraw-Hill International Series.*

Livny, M., Melamed, B. and Tsiolis, A.K., 1992. "The Impact of Autocorrelation on Queuing Systems" *Working Paper????.*

Mireles, J., and Lewis, F.L., 2002. "Blocking Phenomena Analysis for Discrete Event Systems with Failures or Preventive Maintenance Schedules." *Submitted to IEEE trans. on robotics and automation*, 04/04/2002.

Pegden, C.D., Shannon, R.E.,Sadowski, R.P., 1995. "SIMAN: Introduction to Simulation Using SIMAN" (2nd ed.) *McGraw-Hill International Edition.*

Saliby, E., 1989. "Descriptive sampling: a more appropriate approach in Monte Carlo Simulation." *Research Report 86, COPPEAD, Universidade Federal do Rio de Janeiro, Brazil.*

Taha, Hamdy A., 1992. "Simulation with SIMNET II" *SimTec, Inc.*.

Womack, J., and Jones, D.T., 1996. "Lean thinking." *Simon and Schuster.*

PERFORMANCE EVALUATION IN STEEL MANUFACTURING: A CASE STUDY WITH SIMULATION

Adil Bellabdaoui
Faculté Polytechnique de Mons
9, rue de Houdain
B-7000 Mons, Belgium
E-mail: Adil.Bellabdaoui@fpms.ac.be

KEYWORDS
Manufacturing and Engineering (CIME), Computer-aided analysis, Performance analysis, Discrete simulation, Deterministic.

ABSTRACT

This paper discusses a simulation study of cyclic production systems with a travelling crane. The scope of the system covers the tipping stands of the hot iron into foundry ladle, stands of desulphurization and stands of converters. Therefore, the first aim of this study is the development of a discrete-event simulation model in a steel industry case, and a simulation software Witness is used for the treatment. In the second part of the study, other scenarios are proposed for the extension of this system. Stocks and/or machines will be added, which will give a new dimension to the problem. These scenarios will then be modelled and evaluated to determine the best alternatives and/or enhancements. Then, some numerical results are presented and discussed. Finally, the paper is concluded with some recommendations.

1 INTRODUCTION

Simulation is one of the most efficient tools of complex systems analysis. It is necessary for the resolution of flow optimization problems in manufacturing systems. In steel manufacturing, for example, flows complexity - which is due to the huge dimensions (size, mass and distance) of the systems, the different production paces and the different material-handling considerations - makes it difficult to detect possible bottlenecks. It is, therefore; necessary to develop a model of simulation which provides excellent support to real-time, dynamic and adaptable scheduling. Simulation is thought to be the best tool used for any non-trivial, real world system and it plays the role of mediator in a dynamic situation. Many discussions of modelling concepts and steps of simulation can be found in (Shannon 1975; Pidd 1992; Law and Kelton 1999; Law and McComas 1999; Sargent 2001; and Ingalls 2002).

This study presents a simulation model of cyclic production systems with a travelling crane in steel manufacturing. Hence, the role of the travelling crane is to serve the products which require a series of unloading, moving and loading operations. Therefore, the optimal cyclic schedule is defined to determine the best sequence of the travelling crane movements. The present system faces the bottleneck problems. Thus, other scenarios, with Stocks and/or machines will be added, are proposed to the extension of this system. Henceforward, this study suggests the best sequence of the travelling crane and the best alternative to the system in order to reach the hoped production.

In steel industry, among of the majority of manufacturing systems, the crane scheduling is presented in several papers using methods of mathematical model, heuristic algorithms and solution approaches. For instance, a similar study of crane movement has been conducted by J. Bekker (Bekker 2001) in the casting bay at a large South African steel manufacturing. Input, data, assumptions, verification and validation of the simulation model are also discussed. In a second study, facility layout changes are evaluated and the results are compared to the initial operational scenario. The studies (Hansen and Kristensen 2003a; 2003b; 2003c) considers the problem of crane scheduling for plate storage at Odense Steel shipyard. Two cranes are considered to carry out the movements of plates into, around and out of the storage. The problem is solved with a simulated annealing algorithm. The paper (Moon and Hrymak 1999) determines the optimal movement plan for satisfying high usage of shared equipment (like crane) among parallel bases. A novel mixed-integer linear programming (MILP) model for the scheduling of the bath annealing process is proposed. (Bellabdaoui and Allet 2004) discusses a modelling and simulation of one application of steelmaking continuous casing by Arcelor group in Liege, Belgium, where some crane travelling and trucks are managing the flux of production.

The second section overviews the operation of the process under study. Section III introduces the development of the simulation model which steps are described in (Banks 2000) and (Ait Hssain 2000). Section IV proposes some scenarios for the extension of the system. Some numerical results are presented and discussed in section V. Finally, section VI summarizes our work and concludes it with some recommendations.

2 DESCRIPTION OF THE PROBLEM

2.1 Processus description

The iron and steel production consists of three major manufacturing processes: ironmaking, steel-making and rolling. Steelmaking refines pig iron into steel and casts it

into slabs, blooms or billets. Our target is the steelmaking process illustrated in figure 1. It consists of 3 major steps: 2 identical tipping stands ([T]) T1 and T2 (i.e. a wagon is poured into a foundry ladle). 2 stands of desulphurization ([D]) D1 and D2 (i.e. they eliminate the sulphur in a foundry ladle). 3 stands of converters ([C]) C1, C2 and C3 (i.e. unwanted substances, such as carbon and residues, are burnt on through injections of pure oxygen, to produce crude steel). A travelling crane transports the products into the layout: it transfers the foundry ladle from the tipping stands towards the desulphurization stands, it moves from the desulphurization stands into the converters and it goes immediately back to the tipping stands.

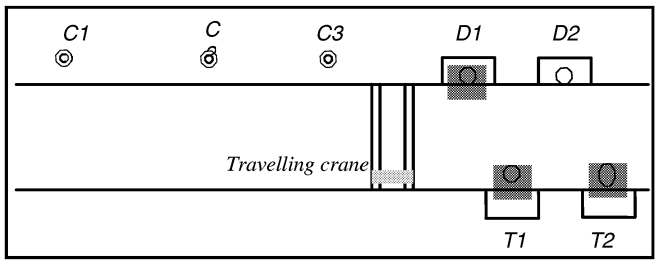


Figure 1: Layout studied

According to figure 2, a few elements are defined below:
Stock-wg. It describes the entrance stock which the wagon parks temporarily.
Tipping stands. By order, the wagons are transferred, one by one, above one of both tipping stands pits. Both stands have two cycles : the wagon tumbling cycle in a foundry ladle, and the wagon restriking cycle.
Desulphurization stands. By order, the travelling crane moves from one of the tipping stands towards one of the desulphurization stands.
Converters. Each converter has four cycles: furnace charging, air blowing, emptying steel and emptying scoria. During its life cycle, the converter is regularly maintained: spout cleaning, pouring gate cleaning, interior gunite and tuyere drilling.
Stock-pa. It describes the exit stock where are stored the steel ladles.

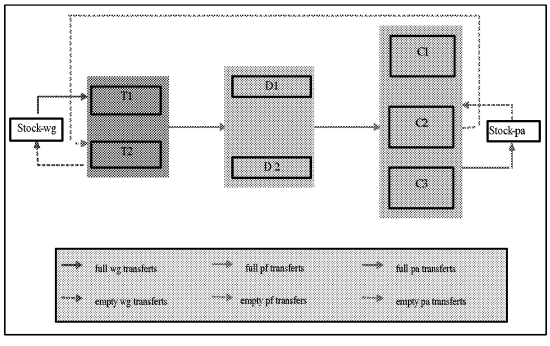


Figure 2: A general scheme for production flux

2.2 Operations principal

In the current configuration of the layout, two over three converters are active (suppose it C1 and C2), the layout has, in maximum, 3 foundry ladle products because it has 4 places for foundry ladle storage (2 in the tipping stands and 2 in the desulphurization stands). If the layout has 4 foundry ladles, we have a block system.
If we suppose that the system is in normal situation (i.e. the two converters are available and there is no maintenance or setup), this configuration allows one and a unique cycle for the travelling crane which it can be repeated indefinitely (figure 3).

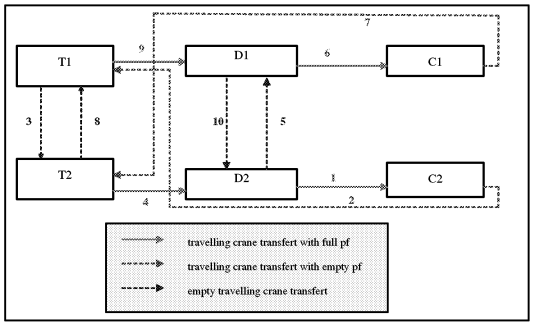


Figure 3: Transfer of the travelling crane in the current configuration

The system starts with a initial state and it returns after each cycle to the same state (i.e. the same machines are loaded, others ones are idle and the travelling crane position is restored). One example of initial state consists in a ready part on each D1, D2 and T2 and no part on T1. When one of the two ladles has finished on D1 or D2 (suppose D2) and one of the two converters C1 or C2 (suppose C2) is idle, the travelling crane transferred the full foundry ladle from D2 into C2 (movement 1) and goes immediately back to T1 with the foundry ladle empty, it moves from T1 to T2 empty and it transferred the full foundry ladle from T2 to D2 (movement 2). The travelling crane moves from D2 to D1 empty and it transfers the full foundry from D2 to C1 (movement 6) and comes back to T2 with the ladle empty. It moves empty from T2 to T1 and it transferred the ladle from T1 to D1 (movement 9) and finally it moves empty to D2. It is obvious that it returns to its initial state.

To simplify the representation of these transfers of the travelling crane, we defined just the travels with full foundry ladle (see figure 4). Therefore, we can conclude that the rest of other transfers of the travelling crane.

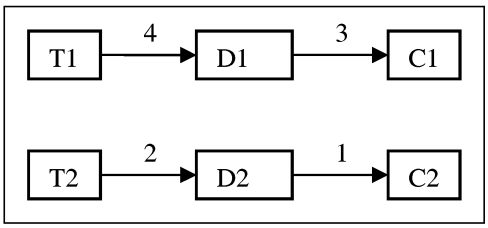


Figure 4: Transfer of the travelling crane in the current configuration

2.3 The bottleneck problem

Finding the bottleneck is not always a trivial task. Currently, there is a number of systematic methods available to find the bottleneck for production systems. The most frequently

methods is measuring either the waiting time or the utilization time (Roser et al. 2003). Our approach measures the utilization of the different machines of the production system. The machine with the highest utilization is considered to be the bottleneck (Brissard and Polizzi 1990).
In the system study, the flux of production is made up of three types of machines, which duplicate in two quantity, to mass-produce alone product. The time of treatment in each machine is presented in Table 1:

Table 1: Data of the problem

Tipping	Desulphurizatio n	Converter
18	17	44

For a slot of one work break of 8 hours (480 min), the capacity per break is defined as follow:

Table 2: Capacity and rate of charge

Machines	Capacity	Charge	Rate of charge
T	54	22	40.74%
D	57	22	38.59%
C	22	22	100%

The bottleneck is the machine converters which can produce just 22 ladles for one work break.
Other probably bottleneck is the travelling crane which it transfers the part between all the machines. Our objective is to validate the bottleneck with simulation model and to propose other scenario for the extension of the system.

3 TYPING REQUIREMENT

3.1 Modelling

The model of simulation has been built by using the software package Witness. For other information for this plate form, the articles (Witness 1999; Rawles 1998; and Mehta and Rawles 1999) can be consulted.

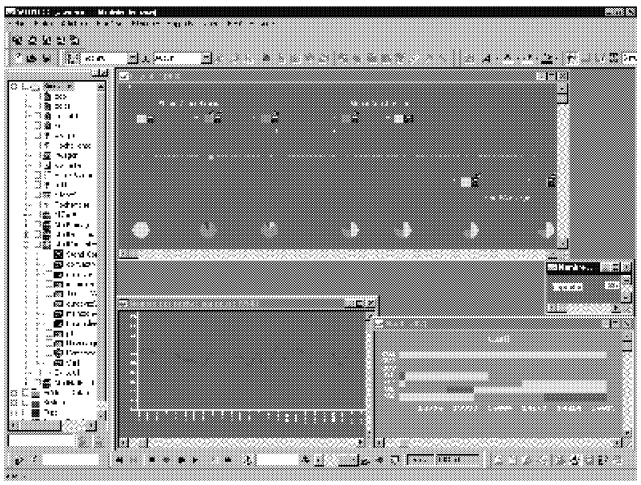


Figure 5: Simulation model of the layout

Entity: main elements were the parts *wg, pf, pa* and an imaginary part *fictif.*

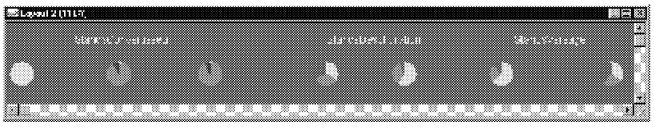
Stock: the stocks representing *Stock-wg* for supply and *Stock-pa* for evacuation.
Attribute: there are some elements which represent a characteristic of an individual part. For example, we use an attribute to characterize weight (*poids*), capacity (*cap*), number (*num*) of ladle.
Machine: we consider three types of machines:

- The first is Tipping Stands which duplicate in two quantities (T1 and T2). Each stand has two cycles, one is the overbalance, the other is holding up.
- The second is a simple machine of Desulphurization which duplicates in two quantities (D1 and D2).
- The third is Converters stands which duplicate in three quantities (C1, C2 and C3). Each stand has 4 cycles with maintenance.

Vehicle: the unique vehicle is the travelling crane which transfers the ladle between various machines.
Track:. there are many tracks on which the travelling crane can transfer the ladle.
Decision variable: there are different types of variables. Some of them are necessary to describe the constraints of the machines; *inter* presents the quantity that we owe emptied from a wagon into the foundry ladle (in the module of tipping stands). Other variables can highlight the results; for example, the number of ladles of one work break (*nbpoche*), the camembert's of the utilization time, the idle time, the breakdown time and the blocking time for the equipment.
The model was verified and statistical reports confirmed that the model gave an accurate representation of the real system.

3.2 Simulation results

The model of simulation starts with three parts, each one is in T2, D1 and T1, and the travelling crane is moved to unload the part in C2 and going back to T2. The model was run for a simulated time of 10 days.
The model of simulation collects an important number of results concerning the utilization of the different machines (Figure 6)



Figures 6: Camembert's of the utilization of the equipment

The most important information is presented in the Gantt chart (Figure 7). This chart shows the time that every machine and the travelling crane have remained at different positions in the process. Gantt chart graphs can be exhibited during and after a run.

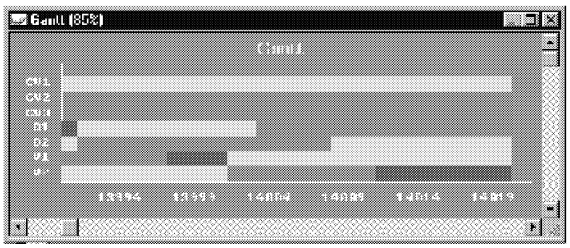


Figure 7: Gantt diagram

We find that the throughput of the system reaches 21 ladles for one break work of operation of simulation; we can see that the number of ladles is between 14, and 22 in figure 8 bellow:

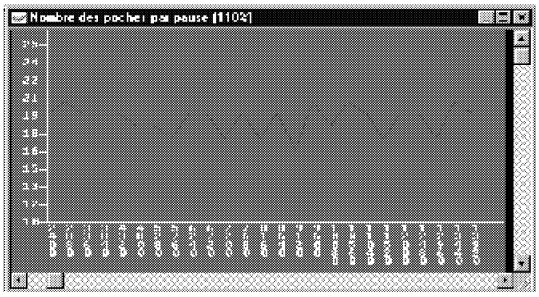


Figure 8: The number of ladles of one work break

According to the figure 6, t is obvious that the machines converters have the largest utilization time for the equipments (more then 88%); therefore, we can confirm that one of the bottlenecks consists of the machines converters.

4 DIFFERENT STRATEGIES OF LAYOUT CONFIGURATION

In this section, we present different scenarios with some hypothesis about materiel resources and the management of the system. In these scenarios we will add stocks and/or machines of the current configuration system (scenario 1), which are presented in Figure 9.
These scenarios are compatible with the actual operating constraints of the layout. The addition of stock means that we can put the ladles pf on the flower and the addition of machine means that we can work with the three machines of converters C1, C2 and C3.
Then, these scenarios will be modelled and we show the possible effect of system configuration change on the production volume.

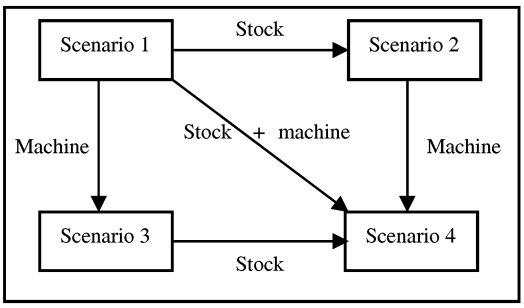


Figure 9: Different scenarios

4.1 Scenario 2

In this scenario we add a stock fictive just near the stands of Tipping in the layout. We can work with four ladles instead of three ladles for the scenario 1.

4.2 Scenario 3

In this scenario we add the real machine C1 in the layout, for the moment we can produce the ladle of steel with the three converters. In this case, we can have just 3 ladles in the layout; the same as in the scenario 1.

4.3 Scenario 4

In this scenario, we add the stock fictive and the machine C1. We work with four ladles.

5 OUTPUT ANALYSIS AND RESULTS

Each scenario defines one or more static cycles for the transfer of the travelling crane, for example, scenario 2 has two cycles and scenario 4 has four cycles. In this paragraph, we choose only one cycle by scenario, we implement the cycles which favours the stands of tipping.

Comparative analysis of the output of the developed scenarios was used to answer the what-if questions stemmed from observation of results of current system.

- General result:

Table 3: A general result of the simulation

	Sc. 1	Sc. 2	Sc. 3	Sc. 4
Number of ladles	580	575	**731**	683
Average number (8h)	19.33	19.16	**24.36**	22.76

In table 3, the single cycle of scenario 3 can produce more then 24 ladles for one work break, and it gives us the best output among the selected cycles.

- Result for the equipment's:

Table 3: Result of the utilization time for the equipment

	Sc. 1	Sc. 2	Sc. 3	Sc. 4
T1	24.99	47.27	31.53	41.49
T2	24.97	2.57	31.45	17.40
D1	34.24	33.98	43.10	40.26
D2	34.24	34.00	43.21	40.38
C1	0.00	0.00	74.25	69.36
C2	88.38	87.66	74.34	69.36
C3	88.59	87.86	74.45	69.67

Table 5: Result of the idle time for the equipment

	Sc. 1	Sc. 2	Sc. 3	Sc. 4
T1	62.99	20.08	48.75	17.42
T2	34.60	1.45	48.65	9.39
D1	33.92	31.15	45.15	40.07
D2	58.77	55.51	46.94	42.40
C1	100.00	100.00	19.09	24.81
C2	4.19	4.91	18.99	24.61
C3	3.98	4.71	18.88	23.88

Table 6: Result of the breakdown time and the blocking time for the equipment

	Sc. 1	Sc. 2	Sc. 3	Sc. 4
T1	12.02	32.65	19.72	41.09
T2	40.44	95.98	19.89	73.21
D1	31.85	34.87	11.75	19.67
D2	6.99	10.49	9.85	17.23
C1	0.00	0.00	6.67	5.83
C2	7.43	7.43	6.67	6.03
C3	7.43	7.43	6.67	6.46

The cycles chosen in scenarios 2 and 4 do not seem to give a good result because the converters are not sufficiently used; the rate of use of the machines converters in scenario 2 (87%) is lower in comparison of scenario 1 (88%), (respectively, the rate of the utilization of the machines converters in scenario 4 (69%) is lower in comparison of scenario 3 (74%)).

6 CONCLUDING REMARKS AND FUTURE WORK

In this paper, we compare the average performance of cyclic production system with an overhead travelling crane using a model of simulation.
Two main objectives may characterize the work described in this paper:

- The development of a realistic simulation model of the cyclic production system with the travelling crane. It combined a complete knowledge of the characteristics and the limitations of the work system.
- The use of that simulation model is to test different scenario in order that it will satisfy the demand of the system. The methodology is to give more flexibility to the system including some extensions of the system and determine the best configuration which gives a maximum output of production.

In the future some studies will be considered.

- The first extension is to implement the other cycles of different scenarios to see their impact on the production.
- The second extension is to envisage dynamic cycles for each scenario and to analyze their influence.

REFERENCES

Ait Hssain, A. 2000. *Optimisation des flux de production: Méthodes et simulation*. Dunod, 2000.

Banks, J. 1999. "Introduction to simulation". In *Proceedings of the 1999 Winter Simulation Conference*. P.A. Farrigton, H.B. Nembhard, D.T. Sturrock, and G.W. Evans (Eds.), 7-13.

Bellabdaoui, A. and S. Allet. 2004. "Modelling and simulation in steelmaking from blast furnace to continuous casting : a case study". *Eighteenth Conference on Quantitative Methods for Decision Making*, A meeting place for industry and university, Orbel 18, Brussels, Belgium, 22-23 January, 2004.

Bekker, J. 2001. "Discrete-event simulation in steel manufacturing: a case study". *Proceedings of the International Conference on Competitive Manufacturing* (2001) 469-476, Stellen-bosch, South Africa.

Brissard, J.-L. and M. Polizzi. 1990. *Des outils pour la gestion de production industrielle*. Ed. AFNOR Gestion 1990.

Cockerill Sambre Group. 1987. *La gestion de la production & du flux matières – principes généraux*, Cockerill Sambre Formation Liège, 1987.

Hansen, J. and T.F. Kristensen. 2003a. "Crane scheduling for a plate storage in a shipyard: modelling the problem". Informatics and Mathematical modelling, Technical University of denmark, DTU, 2003.

Hansen, J. and T.F. Kristensen. 2003b. "Crane scheduling for a plate storage in a shipyard : solving the problem". Informatics and Mathematical modelling, Technical University of denmark, DTU, 2003.

Hansen, J. and T.F. Kristensen. 2003c. "Crane scheduling for a plate storage in a shipyard: experiments and results". Informatics and Mathematical modelling, Technical University of denmark, DTU, 2003.

Ingalls, R.G. 2002. "Introduction to simulation". In *Proceedings of the 2002 Winter Simula-tion Conference*, E. Yucesan, C.-H. Chen, J.L. Snowdon and J.M. Charnes (Eds.), 7-16.

Law, A.M. and M.G. McComas. 1999. "Simulation of manufacturing systems". In *Proceedings of the 1999 Winter Simulation Conference*, eds. P.A. Farrigton, H.B. Nembhard, D.T. Sturrock, and G.W. Evans 56-59.

Law, A.M. and W.D. Kelton. 1999. *Simulation Modeling and Analysis*. 3d ed. New York: McGraw-Hill.

Pidd, M. 1992. *Computer Modelling for Discrete Simulation*. Chichester, England: John Wiley.

Maria, A. 1997. "Introduction to modeling and simulation".In *Proceedings of the 1997 Winter Simulation Conference*, eds. S. Andrad´ottir, K.J. Healy, D.H. Withers, and B.L. Nelson 7-13.

Mehta, A. and I. Rawles. 1999. "Business solutions using Witness". In *Proceedings of the 1999 Winter Simulation Conference*, eds. P.A. Farrigton, H.B. Nembhard, D.T. Sturrock, and G.W. Evans 230-233.

Moon, S. and A.N. Hrymak. 1999. "Scheduling of the bath annealing process –deterministic case", *Computers and Chemical Engineering* 23 (1999) 1193-1208.

Rawles, I. 1998. "The witness toolbox - a tutorial". In *Proceedings of the 1998 Winter Simulation Conference*. D.J. Medeiros, E.F. Watson, J.S. Carson and M.S. Manivannam (Eds.), 223-226.

Roser, C.; M. Nakano; and M. Tanaka. 2003. "Comparison of bottleneck detection methods for AGV systems". In *Proceedings of the 2003 Winter Simulation Conference*. Eds. S. Chick, P.J. Sánchez, D. Ferrin and D.J. Morrice 1192-1198.

Sargent, R.G. 2001. "Some approaches and paradigms for verifying and validating simulation models". In *Proceedings of the 2001 Winter Simulation Conference*. B.A. Peters, J.S. Smith, D.J. Medeiros, and M.W. Rohrer (Eds.), 106-114.

Shannon, R.E. 1975. *Systems Simulation: the art and science*. Englewood Clis, New Jersey, Prentice-Hall, Inc.

Witness. 1999. *Manuel d'utilisation-Version 9.3*. Lanner Group, 1999.

Multilayer Agent-Based Model for Decision Support System Using Psychological Structure and Emotional states

Joël Colloc and Cyrille Bertelle
Laboratoire d'Informatique du Havre
25, rue Philippe Lebon, B.P. 540
76058 Le Havre Cedex France

joel.colloc@univ-lehavre.fr cyrille.bertelle@univ-lehavre.fr

Abstract— **Human decision in complex domains like management, medicine, environment… is stressed by the time and many emotional or psychological constraints that are seldom included in Decision Support Systems (DSS). In a previous article, we presented a Multi-Agent Decision Support System (MADSS) providing the integration of the different categories of knowledge representation. and bringing them to cooperate in elaborating the decision. However, in stressfull domains, for a more realistic system, we need to model as well the impact of emotion and the psychological structure of each actor making a decision. Our model includes a psychological structure layer, an emotion layer, a behavior layer and a decision layer. These layers are all influencing the ability of an agent to make or not an appropriate decision. The emotion equilibriums are represented by attributes. The psychological structure layer triggers tasks and behaviors according to the emotion states. Automata are designed to describe individuals typical behaviors, abilities or handicaps according to different psychisms and personalities. The behavior layer set a current goal, selects a strategy or a scenario to achieve it. The decision layer makes a decision if the agent is able to provide it.**

1. INTRODUCTION

Some Decision Support System (DSS) have been proposed to help workers during their job to cope with the steps of the decision making in complex domain like management, medicine and environment. These DSS are provided for enhancing the efficiency of the decisions in these domains. Therefore, this kind of DSSs is well suited to aid a user to solve a problem and take an appropriate decision according to available knowledge sources or stored experience[9][11].

However, these DSSs have no representation of each actor's ability to take appropriate decisions. Moreover, these DSS are not relevant when more or less emotional agent showing different psychological structure are interacting in the environment and are either making suitable or poor decision. The purpose of this work is to provide a Multi-Agent Decision Support System (MADSS) to model the emotion and the psychological structure of interacting actors making decisions in complex domains and stressfull contexts. This model will be used to build simulators of agent attitude in various situations such as emergency medicine, trading, education, environment and industrial catastrophes, military conflicts… In all these domains, modelling the emotional aspect and the psychological structure of the agents is essential. Many approaches were proposed to model the agent emotion for different purposes: providing more human, friendly user interface and facial animation [1][8], educational games and pupil metacognition [3] [5], military conflicts [13].

Firstly, different existing models of emotion are briefly reminded. Our main contribution is to introduce the representation of the individual's psychological structure which is closely interacting with the emotion level.

Thus, the section 2 describes a multilayer model composed of the psychological structure layer, the emotional state layer, the strategy and behavior layer, the decision making layer. A schema describes the interactions between these different layers and the decision process.

The section 3 describes the psychological structure layer. It firstly provides a generic model of typical neurotic tendencies and common personality structures. The model is then specialised to fit with different kinds of neurosis.

The section 4 presents the emotional state layer which relies on the well known Ortony, Clore and Collins' model (OCC model) [6][7]. The emotion state layer is composed of balance attributes defining the emotional state of each agent.

The section 5 shows how the emotion and the psychological structure are modelised and combined together to provide the agent strategie and behavior.

The section 6 describes the decision making layer defining the appropriate tasks to achieve a goal and the relationship with the strategies and behaviors layer. This latter determines if the agent emotional state allows the agent to really perform the required tasks provided by the decision making layer. At last our approach is compared with other similar works. The conclusion section summarizes the main contribution of our model and the perspectives of this work.

2. A MULTILAYER MODEL OF MULTI-AGENT DECISION SUPPORT SYSTEM

Most available computationally models of emotion rely on the OCC model developed in 1988 [6]. The OCC model defines events, agents and objects. Events are considered to induce emotional consequences. Agents are able of actions that have effects on the environment. Objects have imputed properties. The OCC model represents emotions as valenced reactions to the perception of the world. That is: one can be pleased about the consequences of an event or not (pleased/ displeased); one can endorse or reject the actions of an agent (approve/disapprove) or one can like or not aspects of an object (like/dislike). Then, the events can have consequences for others or for oneself and on acting agents. Thus, the different emotional balances are depicted by couples of (positive/negative) reactions represented by variables.

		+	-
Consequences of events	For others	Happy for	Resentment
		Gloating	Pity
	For self	Hope	Fear
		Joy	Distress
Actions of Agents	Self Agent	Pride	Shame
		Gratification	Remorse
		Gratitude	Anger
	Other Agent	Admiration	Reproach
		Gratification	Remorse
		Gratitude	Anger
Aspects of Objects		Love	Hate

Table 1 Couples of (positive/negative) emotion variables according to the OCC model

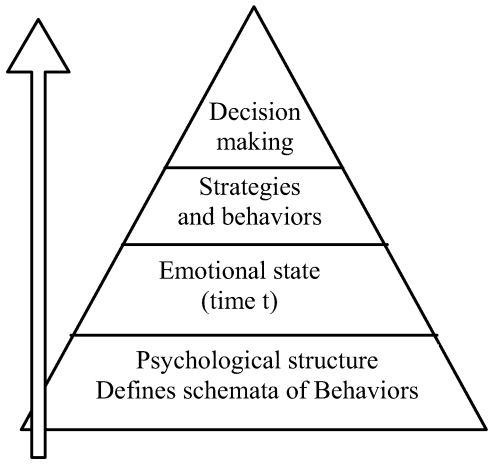


Fig. 1 A multilayer model of decision

For short, we do not provide the specialization tree of the OCC model but we just summarize the couples of variables in table 1.

A complete description of the OCC model is available in [6] and computational applications in [7][4][1].

The aim of our proposition about model of decision is to use a multilayer description with 4 layers which will be explained in the following paragraphs. Note that the base of this model is the psychological structure as described in figure 1.

3. THE PSYCHOLOGICAL STRUCTURE LAYER

First, we describe the psychological structure which are able to act on the decision making behavior which is influence by emotional aspects. In figure 2, we present how and where the psychological structure intervene in the emotional learning loop.

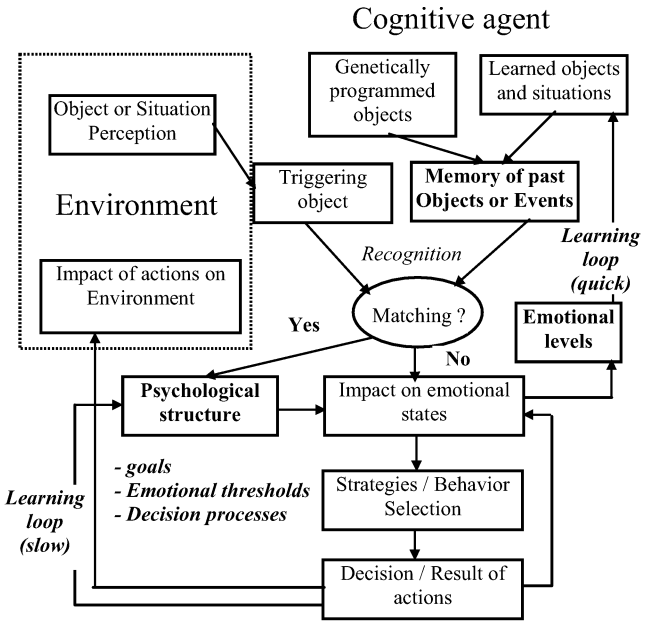


Fig. 2 . The emotion learning loop and psychological learning loop

Psychological structure can be studied through neurotic aspects of the considered personality. We show in the table 2, a description of such personality, while table 3 shows a neurosis classification.

A neurotic personality		
- A list of anxiogeneous events		
	- An anxiogeneous event is :	
	Or	- The perception of a triggering object(s) having specific characteristics (eg. fear of mice)
		- The perception of situation(s) or scanario(s) defining specific contexts (eg. agoraphobia)
- A list of emotional effects		
	- An emotional effect is :	
	And	- The increase or a decrease of emotional parameters
		- The Emotional states should trigger defence behaviors
- A list of defence behaviors		
	And	- The agent sets a goal (most of the time : the elimination of the unpleasing situation or of the object causing the disagreeable emotional state
		- The agent elaborates a defence strategy. Each strategy is a set of connected actions executed to achieve the goal (destroy the object, flight, avoid, prevent…).
	Or	- The agent may show inappropriate behaviors caused by the pathological neurotic personality

Table 2. A generic model of the neurotic personality

Neurosis	Triggering object	Anxiogeneous events	Emotional effects	Defence behavior
Obsessional neurosis	Lack of objects Lack of misdoing	Anxiety of loosing an object. Anxiety of not achieving appropriate tasks	Severe inner struggle Fear and anxiety	Ritual and necessity to carry out actions again and again Compulsive need to verify
Phobia neurosis	Animals Crowd Closed places	Abnormally intense dread of certain objets or specific situations	Dread Fear	Flight Avoiding Prevent
Hysterical neurosis	Herself or himself image Others' sight	Anxiety of offending or displeasing	Anxiety Halluci-nations	Repression Flight Avoiding indifference Versatility Theatrical behavior

Table 3. A simplified neurosis classification

4. THE EMOTIONAL STATE LAYER

In this section, we describe the emotional state layer of the model. This layer relies on the Ortony Clore Collins (OCC) model of appraisal which is the most widely accepted. The OCC model defines a set of 22 emotion parameters representing a certain positive and negative intensity of 11 items. The emotion quickly changes over time.

We use a sigmoid fuzzy logic function to describe each couple of parameters of the OCC model table1. The figure 3 depicts the example of Joy/distress concerning the effect of an event k evaluated by a score [-1,1] that is considered as desirable when is near to 1 or undesirable when it is near to -1, neutral when it is near 0. All other couples of variables of the OCC classification table 1 are represented in a similar way. The item k could represent the consequences of an event, the impact of agent actions for self or others and the aspects of an object.

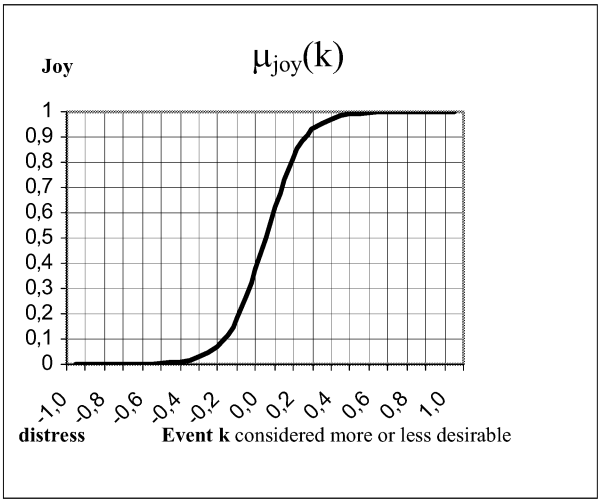


fig. 3 A fuzzy logic membership function μjoy(k)

5. THE STRATEGIES AND BEHAVIORS LAYER

Formalism proposed

This layer has to manage previous layers, psychological level and emotional states and to define behaviors taking into account them.

The formalism proposed is based on finite-state automata composed of:

- a set of internal states characterizing mental situation, for example;
- transitions which describe the way going from one state to another. In the following, the event k evaluated by the score in the interval [-1,1] is the shutter release of the transitions.

We proposed to complete this formalism allowing us to introduce stochastic aspects in behavior descriptors. For this, each transition has to output a probability which indicates the

chance that the transition was really done. To achieve this aspect, we base our model on automata with multiplicities or probabilistic automata as a sub-class of them.

Thus, the mathematical formalism for the behavior representation at each time t is a probabilistic automata $A(t)$ as the following 5-uple $(\Sigma, Q, I, J, \delta)$:

- Q is the finite set of internal behavior states;
- Σ is the set of perceived event k. As previously indicated, each event k is evaluated by the score in the interval [-1,1]
- I and J are respectively the set of initial and final states from whom a behavior (or specific strategy, for example) can start and finish respectively.
- δ is a mapping from $Q \times \Sigma \times Q$ to [0,1] which describe all transitions associated to a couple of states and a event in input. The output of the mapping is a stochastic value.

Behavior, emotional state and psychologic structure bounding

We explain now, how the mapping δ is linked with emotional states as following:

1. For each transition j and each event k, we define
$$I_j = \arg \bigotimes_{i \in [1,N]} \alpha_{j,i}(t) \mu_i(k)$$
Where N is the number of different emotion parameters (in OCC model, this corresponds to the numerical value: 22), $\alpha_{j,i}(t)$ are relative importances between all emotional parameters at time t for the transition j, the operator $\otimes$ corresponds to one of the two classical operators, $\wedge$ or $\vee$, that is in fact the minimum or the maximum value. As usual, the operator $\arg\otimes$ return the index corresponding to the min or max value .
2. We consider then the fuzzy logic function associated to the previous computed index I_j:
$$E_j(k) = \mu_{I_j}(k)$$
3. We defined a random variable associated to two values u and v which respectively correspond to the fact the transition is crossed or not.
$$X : \begin{cases} [0,1] \rightarrow \{u, v\} \\ if\ x \le E_j(k)\ then\ X = u \\ else\ X = v \end{cases}$$
4. Finally, the probability computed as output of the transition j, associated to an event k is:
$$P_j = P_j(\{X = u\}) = E_j(k)$$

We have proposed a computable formalism (automata-based) for modeling the behavior in respect of the emotional aspect. One of the major aspects in this operating automata description is the introduction of the coefficients $\alpha_{j,i}(t)$ for each transition j, each emotion parameter I at the instant t. These coefficients are obtained from the psychological structure and a database knowledge about them. So, via these parameters, the psychological structure acts on the behavior model, taking into account nevrotic personality, for example. Another aspect of the efficiency of these parameters is based on the fact that they can evolve, using genetic processus. This aspect can confer to the behavior some adaptive aspects.

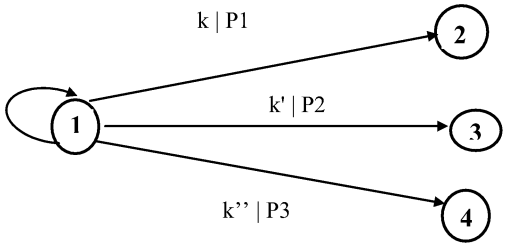


fig. 4 A probabilist automaton for modelling the emotion / behavior relationship in a specific psychological structure

Illustrative sample

Let us take a simple situation: "buying an object", eg. a computer. In this general type of problems, one can summarize different choice criteria according different aspects : the knowledge concerning the object (the features, capabilities, price …), the feeling for the object and the personality which is built according to the life experience. Thus, the first part of the decision process is supported by reasoning on the available information concerning the object to buy. The second part is purely driven by the emotion and feeling of the object: the good or bad perception (shape, color, smell,…) and moreover, the relation with pleasant or unpleasant past events or perceptions previously stored in the memory. The third part is defined by the psychological structure and the corresponding decision schema which are evolving during each individual's life. Therefore, our model is composed of three so called closely interelated layers: psychological, emotional state, decision making.

The psychological layer is defined by a library of general behavior and strategies automata. These automata define meta-schema according to the current psychological knowledge. They are driving the agent general behavior and strategy.

The emotional state layer is composed of a set of valanced emotional state variables determining the issues of transitions of behavior and strategies automata in the previous layer.

The decision making layer is represented by a library of knowledge automata defining the correct decision process to solve a specific problem. The stored knowledge represents the appropriate solving method and the relevant data sources concerning a class of problems. Even if the relevant knowledge is available, this layer don't take into account the agent emotional or psychological ability to actually solve the problem at the moment. The following table summarizes the necessary steps to buy the object. Figure 5. represents the transition according two different psychological structure: optimistic or obsessionnal.

	Tasks to choose and buy an object
1	Set a list of wished object properties : preferences
2	Select a set of objects showing some relevant properties in a group of available objects
3	Get the specific features of selected objects
4	Compare the features of the selected objects with the required properties
5	Rank the selected object according to the preferences
6	Verify the properties of the best ranked objects
7	Decide to buy or reject an object
8	Pay for the object

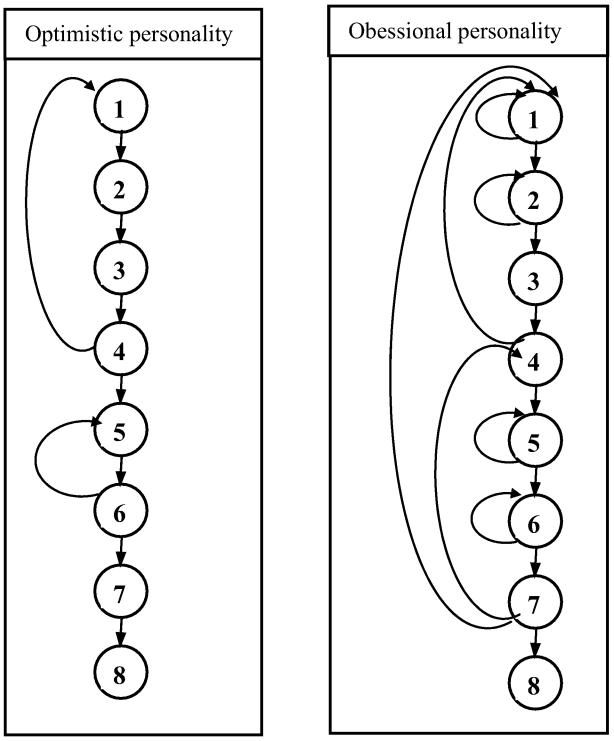


Fig. 5 Automata of two psychological structures during the same buying process

6. THE DECISION MAKING LAYER

This layer is mainly described in a previous article [10]. We briefly summarize the architecture of a Multi-Agent Decision System which provide the necessary knowledge sources to take appropriate decisions but don't take into account the emotion and the psychological structure of the decision-maker.

A MADSS is using four agent categories that are specialized from a generic type (GCAT) described in [10]. A first specialization provides *Knowledge Model Agent Types* (KMAT), which are specific according to the knowledge model (rules, evaluation functions, Case-Based Reasoning…) but independent to the application purpose. The second specialization supplies *Domain Specific Agent Types* (DSAT), which inherit reasoning and knowledge model capabilities from the appropriate KMAT and apply them to the specific domain e.g. in medicine: (infectious disease diagnosis, poisoning prognosis, epilepsy therapy, patient and treatment follow-up, previous clinical case retrieval…)… Then, Task Specialized Agents (TSA) are instantiated from DSAT and commit to tasks elected by the supervisor agent that coordinates the whole decision process. The TSAs access the domain relevant bases containing knowledge and data (represented with the appropriate model) when necessary.

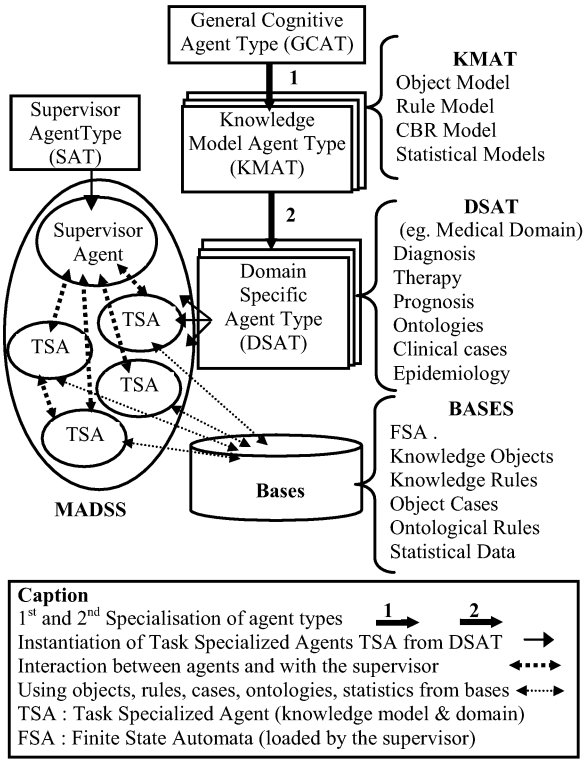


Fig. 6 : Specialisation of agents involved in a Multi-Agent Decision Support System (MADSS)

The ontological agent type is a DSAT, which formalizes the terminology, and the entity definitions used during the information transaction between agents [16] [17].

The supervisor agent type provides a unique instance named *the supervisor agent* that make an intensive use of finite state automata which define the necessary tasks to build the appropriate decision. The decision-making layer is consistent with the other layers of the model defining the emotion and the psychological structure of the decision-maker.

Thus, the model both represents the necessary decision steps, the psychological and emotional ability of the decision-maker to actually take the appropriate decision. Such a model is much more suited to realistically simulate the agent attitude and behavior in stressful situations or when the agent's personality hinders to take a decision.

7. PERSPECTIVE TO MODELISATION AND SIMULATION FOR MEDICAL DECISION MAKING

We are currently using our model to support medical decision making for all stakeholders. The choice of the treatment strategy depends on the prognosis, the age, the mental state and the physical state of the patient. Moreover, patients are experiencing emotions (mood, feelings, fear, …) that influence the result of the treatment. Therefore, our model is well suited to incorporate emotional and time aspect in medical decision support system.

8. CONCLUSION

We have proposed an approach of multi-agent decision support systems able to deal with the emotion and the pychological structure of the agents. The main contribution of this work is to provide a general multilayer framework of interacting agent emotion and psychological structure. This model relies on the OCC model, and fuzzy logic membership functions defining a mental state. According to the value of the mental state, the agent's behavior is driven by the transitions of a probabilistic automaton. Therefore, the transitions define if the emotion state is compatible or not with the execution of an action and they express the probability for the agent to actually carry out this action. Thus, the automaton represents the agent's psychological structure of his behavior according to the emotion state which itself modified by the occurrence of pleasant or unpleasant events.

9. REFERENCES

[1] S. Kshirsagar, N. Magnenat-Thalmann ,"a Multilayer Personality Model",Proceedings of the 2nd international symposium on Smart graphics table of contents, ed. Hawthorne, New York, ACM Press, NY, USA, 2002, pp.107-115, ISBN:1-58113-555-6

[2]E. André, M. Klesen, P. Gebhard, S. Allen, and T. Rist, «Exploiting Models of Personality and Emotions to Control the Behavior of Animated Interactive Agents » in J. Rickel et al. Eds., Proceedings of the 4th Int. Conference on Autonomous Agents, pp 3-7, Barcelona.

[3] C. Conati « Probabilistic Assessment of User's Emotions in Educational Games » Journal of Applied Artificial Intelligence, special issue on " Merging Cognition and Affect in HCI", 2000, vol 16 (7-8), pp. 555-575.

[4] M.S. El-Nasr, J. Yen, T.R. Ioerger FLAME - Fuzzy Logic Adaptive Model of Emotions », Autonomous Agents and Multi-agent Systems, Autonomous agents and multi-agent systems, 2000, no3, pp.219-257.

[5] J. Faivre, C. Frasson, R. Nkambou, « Gestion émotionnelle dans les systèmes tuteurs intelligents », TICE 2002, Lyon, 2002, pp. 73-80.

[6] A. Ortony, G. Clore, A. Collins, « The cognitive structure of emotions », Cambridge University press, 1988, Cambridge, MA.

[7] R.W. Picard, « Affective computing », the MIT Press, Cambridge, MA, 2nd ed., MIT USA, 292p., ISBN 0-262-161170-2

[8] R.C. Hubal, G.A. Frank, C.I. Guinn, « Lessons Learned in Modeling Schizophrenic and Depressed Responsive Virtual Human for Training », Int. Conf. on Intelligent Interfaces, 2003.

[9] L. Padgham , G. Taylor, « A System for Modelling Agents having Emotion and Personality » ; PRICAI 1997, Workshop on Intelligent Agent Systems.

[10] J. Colloc J., C. Sybord, « A Multi-Agent Approach to Involve Multiple Knowledge Models and the Case Base Reasoning Approach in Decision Support Systems, In proceedings of 35th IEEE Southeastern Symposium on System Theory SSST'03, WVU, Morgantown USA, 2003

[11] J. Colloc, L. Bouzidi, "A Case Based Reasoning Decision Support System for use in Medicine", UPGRADE Vol. II, n°1, Feb 2001, pp30-35.

[12] I.J. Roseman, P.E. Jose, M.S., Spindel, "Appraisals of emotion-eliciting events: testing a theory of discrete emotions". Journal of Personality and Social Psychology, vol.59, no 5, pp. 899-915.

[13] M. Johns, B.G. Silverman, "How Emotions and Personality Effect the Utility of Alternative Decisions: A Terrorist Target Selection Case Study",

[14] N. R Jennings., "On agent-based software engineering", *Artificial Intelligence, Elsevier,* vol. 117, no 2,, 2000, pp. 277-296.

[15] J. P. Müller, "The Design of Intelligent Agents A layered Approach", *Lecture Notes in Computer Sciences, Lecture Notes in Artificial Intelligence, Springer Verlag* vol. 1177, Berlin: 1996. p.227.

[16] S. Falasconi, G. Lanzola, M. Stefanelli., "An Ontology-Based Multi-Agent Architecture For Distributed Health-Care Information Systems", *Methods of Information inMedicine, F.K. Schattauer Verlagsgesellschaft Mbh*, 1997, pp. 20-29

[17] T.R. Gruber, " A translation approach to portable ontology specifications ", *Knowledge Acquisition*, vol.5. no 2,1993, pp. 199-220

[18] K Sycara and Z. Zeng, "Multi-Agent Integration of Information Gathering and Decision Support", *in Proceedings of the 1996 12th European Conference on Artificial Intelligence ECAI'96, ed. By W. Wahlster, John Wiley & Sons*, pp. 549-553.

OTHER PAPERS

COMPARISON OF MODELS TO DESCRIBE CRYSTALLIZATION KINETICS OF FATS

Imogen Foubert[a,b], Peter A. Vanrolleghem[b] and Koen Dewettinck[a]
[a] Food Technology and Engineering
[b] BIOMATH
Ghent University
Coupure Links 653
B-9000 Ghent, Belgium
E-mail: Imogen.Foubert@UGent.be

KEYWORDS
fat, lipid, crystallization, kinetics, mathematical modelling, model selection, asymmetry

ABSTRACT

Insight in the crystallization kinetics of fats, i.e. when and to what extent fat components crystallize under certain conditions, is important for controlling operations in the manufacturing process which are based on fat crystallization. Quite some studies have been published in which the differences in the crystallization behaviour between different products and crystallization conditions are quantified using a mathematical model to describe the isothermal crystallization kinetics of fats. Four models (Avrami with and without induction time, Gompertz and Foubert) were compared. The fit of the Gompertz and Foubert models is much better than that of the Avrami model (with and without induction time). The fit of the Foubert model is also significantly better than that of the Gompertz model for most data series. This can be explained by the fact that the Foubert model offers flexibility concerning the asymmetry of the curve while the Gompertz model has a fixed asymmetry. The errors on the parameter estimates of the Gompertz and Foubert models are of equal amplitude but lower than those of the Avrami model.

INTRODUCTION

Fats are being crystallized for various reasons, such as fractionation into certain groups of triglycerides with varying melting and physical properties or to give food products a certain texture. Products in which fat crystallization is important include chocolates and confectionary coatings, dairy products such as butter and cream and vegetable spreads (margarine) (Hartel 1992).
The kinetics of fat crystallization, being dependent on the composition and on the processing conditions, is important for controlling operations in the food industry to produce the desired product characteristics (Metin and Hartel 1998). Modelling the kinetics of fat crystallization is used to gain insight in, quantify and predict effects of fat composition, processing conditions, additives and minor components, …
The Avrami model (Avrami 1939 and Avrami 1940) is the most frequently used model to describe the isothermal crystallization kinetics of fats. It was initially developed for metals, later it was also used a lot for polymers and fats. The original Avrami model (equation (1)) can be adapted to account for the fact that not necessarily all fat is crystallized at equilibrium. An extra parameter a_A [%], representing the maximum amount of crystallization, is therefore added to the model leading to equation (2):

$$X(t) = 1 - e^{-kt^m} \quad (1)$$

$$f(t) = a_A \times \left(1 - e^{-kt^m}\right) \quad (2)$$

where X [-] is the relative amount of crystallized fat, represented as a number between 0 and 1, f is the amount of solid fat at time t [%], k is a crystallization rate constant which depends primarily on the crystallization temperature [h^{-m}] and m is the Avrami exponent [-], which is a combined function of the time dependence of nucleation and the number of dimensions in which growth takes place.
It is also possible to add a fourth parameter to the Avrami model to account for an induction time t_ind_A [h] (Lambrigger 1996). The model then changes to:

$$f(t) = a_A \times \left(1 - e^{-k(t - t_ind_A)^m}\right) \quad (3)$$

A few authors have used a reparameterized Gompertz model to describe the isothermal crystallization kinetics of fats (Zwietering 1990):

$$f(t) = a_G \times \exp\left\{-\exp\left[\frac{\mu_G \times e}{a_G} \times (\lambda_G - t) + 1\right]\right\} \quad (4)$$

where a_G is the value for f as t approaches infinity [%], μ_G is the maximum increase rate in crystallization [%/h] (or the tangent to the inflection point of the crystallization curve) and λ_G is a measure for the induction time [h] defined as the intercept of the tangent at the inflection point with the time-axis. The Gompertz model originates from demography but was mainly used to describe bacterial growth.
Foubert et al. (2002) have presented a model describing the fat crystallization process as a first order forward reaction compensated by a reverse reaction of order n. It is written in terms of a variable h [-], the relative amount of remaining crystallizable fat:

$$h = \frac{a_F - f}{a_F} \quad (5)$$

where a_F [%] is the value of the variable f [%] for t [h] approaching infinity. In contrast to f, which increases in time in a sigmoid way, this variable h is related to the remaining supersaturation (i.e. the driving force for crystallization) and thus decreases in time in a sigmoid way. The model can then be written, in the form of a differential equation, as:

$$\frac{dh}{dt} = K \times (h^n - h) \qquad (6)$$

$$h(0) = \frac{a_F - f(0)}{a_F}$$

where K is the rate constant [h^{-1}], n the order of the reverse reaction [-] and $f(0)$ the initially present amount of crystals [%].

To simplify parameter estimation the model was converted to its algebraic solution, assuming isothermal conditions. Additionally, it was decided to represent the model as function of the induction time t_ind_F instead of $f(0)$ since the physical interpretation of the parameter induction time is more straightforward than that of the parameter $f(0)$ and the induction time can be more easily found back on a crystallization curve and the induction time is an important industrial parameter.

$$f(t) = a_F \times \left[1 - \left[1 + \left((1-F)^{1-n} - 1\right) \times e^{-(1-n) \times K \times (t - t_ind_F)}\right]^{\frac{1}{1-n}}\right] \qquad (7)$$

in which t_ind_F [h] is defined as the time needed to reach F% of crystallization, where F for example could be 0.01.

It is the aim of this paper to compare these different models used to describe the isothermal crystallization kinetics of fats. They are compared using mathematical model selection criteria based on the quality of fit and selection criteria based on checking the assumption of random and independent residuals. Furthermore the error on the parameter estimates and the ability to fit an asymmetric curve of the different models is compared.

MATERIALS AND METHODS

Data series used

The data series used are isothermal crystallization curves of different samples of cocoa butter and milk fat (fractions) measured by differential scanning calorimetry and pulsed nuclear magnetic resonance.

Parameter estimation

The different algebraic models were fitted to the data series by non-linear regression using the Sigmaplot 2000 software (SPSS Inc., Chicago, IL, USA). This software uses the Marquardt-Levenberg algorithm to find the parameters that give the 'best fit' between the model and the data.

Model selection

Several methods exist to evaluate the quality of different models after fitting each model to the data. These methods can be subdivided into information criteria, methods that go back to statistics and techniques in which an analysis is made of the residuals between model predictions and measured data. Vanrolleghem and Dochain (1998) give an overview of the model selection methods. The methods used in this study include the information criteria FPE, AIC, BIC and LILC and the statistical F-test to decide whether the more complex model j is significantly (with a significance level α) better than model i. To check whether the residuals are independent of each other, the run- and auto-correlation tests were applied.

Errors on parameter estimates

The errors on the parameter estimates were calculated by Sigmaplot for each model.

RESULTS AND DISCUSSION

Model selection based on quality of fit

The four models - Avrami with and without induction time, Gompertz and Foubert model - were compared with the mathematical model selection criteria. Table 1 indicates which model is selected on the basis of the different model selection criteria.

Table 1: Number of Data Series (on a total of 53) for which each Model is Selected on the Basis of Different Model Selection Criteria. A = Avrami, Ai = Avrami with induction time, G = Gompertz, F = Foubert

	FPE	AIC	BIC	LILC	F-test	Run test	Auto corr
A	0	0	0	0	0	0	0
Ai	0	0	0	0	0	0	0
F	42	41	42	44	41	39	10
G	11	12	11	9	7	4	1
F/G	0	0	0	0	5	1	8
Ai/F	0	0	0	0	0	7	1
Ai/F/G	0	0	0	0	0	2	3
A/Ai/F	0	0	0	0	0	0	2
All	0	0	0	0	0	0	28

It could be concluded that the Avrami model with induction time fits the data significantly better than the original Avrami model and this for most of the data series. The Gompertz model offers a large improvement compared to the Avrami model. When the Gompertz model is compared with the Avrami model with induction time, the Gompertz model nearly always performs better even though it uses one parameter less. The statistical F-test showed that for all data

series the Foubert model with four parameters performs significantly ($\alpha = 0.01$ except for one data series, $\alpha = 0.05$) better than the Avrami model without induction time having only three parameters, and for most of the data series (41 out of 53) also significantly better than the Gompertz model also only having three parameters.
It could also be concluded that the Mean Sum of Squared Residuals (MSR) value is much more constant for the Foubert model than for the Gompertz model. For some data series the values for the Gompertz model are only slightly higher, and for some data series even slightly lower than for the Foubert model, while for other data series the MSR value for the Gompertz model is much higher than the one for the Foubert model. For the data series where the Foubert model performs worse, the difference between the two values is small. These results indicate that the Foubert model is more flexible in describing the crystallization kinetics of fats. It gives a good fit for all the data series used, while the Gompertz model gives some very good fits as well as some significantly poorer fits.

Model selection based on independence of residuals

Originally the absolute amount of crystallization f was calculated each minute. After parameter estimation, however, it appeared that using this sampling frequency, the residuals of all models were auto-correlated. Auto-correlated residuals lead to underestimated errors on the parameter estimates. A solution to this auto-correlation problem was to subsample the data series, i.e. to use only one data point each five minutes. Using this sampling frequency, for the Foubert model a maximum of one auto-correlation above the threshold was found for all data series. For the other models a limited number of data series showed more than one auto-correlation above the threshold.

Errors on parameter estimates

Table 2 compares the errors on the parameter estimates for the four models. The table shows that the errors are largest for the Avrami model (with and without induction time). The errors on the parameter estimates of the Gompertz and Foubert models are of similar magnitude, except for the error on the n parameter of the Foubert model, which is higher. However, this was to be expected since more parameters have to be estimated from the same data series.

Table 2: Comparison of the Errors [%] on the Parameter Estimates (mean values are reported and parameters with a comparable meaning are reported on the same line)

Avrami		Avrami induction		Gompertz		Foubert	
k	10	k	7.5	μ_G	1.5	K	1.5
m	5	m	7.5				
		t_ind_A	7.5	λ_G	0.75	t_ind_F	1.5
a_A	1	a_A	0.5	a_G	0.25	a_F	0.2
						n	4

Ability to fit asymmetric curves

To gain a clear insight into the differences between the models, their ability to fit an asymmetric curve was tested. The asymmetry of a curve was defined as:

$$asym = \frac{t_{90\%} - t_{50\%}}{t_{50\%} - t_{10\%}} \qquad (8)$$

where $t_{F\%}$ [h] is the time needed to reach F% crystallization. A symmetric curve has a value of 1 for *asym*. Of course, alternatives with other percentages of crystallization are also possible to define the asymmetry of the crystallization curves.
The *asym* value was calculated for the four models. First, the time needed to reach F% of crystallization was calculated as function of the model parameters. The thus obtained formula with different values for F was used to calculate the *asym* value according to equation (8).
For the Avrami models with and without induction time the same expression for *asym* was obtained:

$$asym = \frac{\sqrt[m]{-\ln 0.1} - \sqrt[m]{-\ln 0.5}}{\sqrt[m]{-\ln 0.5} - \sqrt[m]{-\ln 0.9}} \qquad (9)$$

The asymmetry of the Avrami model (with and without induction time) is thus only dependent on the value of the Avrami exponent m. It was calculated that the Avrami model is symmetric for m equal to 3.26. For m values higher than 3.26, the *asym* value is lower than 1, meaning that the start of the crystallization process is slower than the end. The inverse is valid for m values lower than 3.26. The value of 3.26 is independent of the values for F% chosen in the definition of the *asym* value.
When the original Avrami model was fitted to the data series, m values between 3 and 6, coinciding with *asym* values around 1 or smaller, were obtained. This would mean that the start of the crystallization is slower than the end. This is, however, not concordant with the experimental data which may explain why the Avrami model does not provide good fits.
The Avrami model with induction time takes account of the induction time before the start of crystallization. Obtained m values for the Avrami model with induction time coincide with *asym* values larger than 1, which is concordant with the experimental data. This probably explains why the Avrami model with induction time fits the data significantly better than the original Avrami model.
For the Gompertz model the following expression was obtained for the asymmetry:

$$asym = \frac{-\ln(-\ln(0.9)) + \ln(-\ln(0.5))}{-\ln(-\ln(0.5)) + \ln(-\ln(0.1))} = 1.57 \qquad (10)$$

Equation (10) proves that the *asym* value is fixed at 1.57 for the Gompertz model. This is an important feature of this model since it means that it does not offer any flexibility concerning the asymmetry of the curve. This probably explains why the Gompertz model fits certain datasets very well (those showing an *asym* value around 1.57) and others considerably worse. The calculation of the *asym* value can therefore be used as a method of a priori model selection, since the evaluation of the *asym* value of a data series would

allow to decide beforehand whether the Gompertz model will fit well or not.
For the Foubert model the following expression was obtained for the asymmetry:

$$asym = \frac{\ln(0.5^{1-n} - 1) - \ln(0.1^{1-n} - 1)}{\ln(0.9^{1-n} - 1) - \ln(0.5^{1-n} - 1)} \qquad (11)$$

From equation (11) it was deduced that the asymmetry of the Foubert model is only dependent on *n*. For *n* equal to 2, a symmetric curve is obtained. Table 3 gives the *asym* values for some selected values of the model parameter *n*.

Table 3: Relationship between the model parameter *n* of the Foubert model and the asymmetry of the curve

n	asym
1.5	0.81
2	1
3	1.37
4	1.69
5	1.94
6	2.12
7	2.27
8	2.37

An alternative method to define the symmetry of a curve was also explored. A symmetric model was defined as a model for which the *f* value at the inflection point equals 0.5 *a*, meaning that exactly half of the attainable crystallization is already reached at the inflection point. This way of defining the symmetry led to precisely the same results: the Avrami model is symmetric for an *m* value of 3.26, the Gompertz model is never symmetric and the proposed model is symmetric for an *n* value of 2.

CONCLUSIONS

It was proved that the Foubert model always performs better in describing the isothermal crystallization kinetics of fats than the often-used Avrami model (with and without induction time). It also performs better than the Gompertz model for the majority of the data series. The main difference between the Gompertz and the Foubert model is the flexibility of the latter to fit crystallization curves with different degrees of asymmetry.

ACKNOWLEDGEMENTS

Imogen Foubert is Postdoctoral Fellow of the Fund for Scientific Research – Flanders (Belgium) (F.W.O. – Vlaanderen).

REFERENCES

Avrami, M. 1939. "Kinetics of phase change. I. General theory." *Journal of Chemical Physics*, 7, 1103-1112.

Avrami, M. 1940. "Kinetics of phase change. II. Transformation-time relations for random distribution of nuclei." *Journal of Chemical Physics*, 8, 212-224.

Foubert, I.; P.A. Vanrolleghem; B. Vanhoutte; and K. Dewettinck. 2002. "Dynamic mathematical model of the crystallization kinetics of fats." *Food Research International*, 35, 945-956.

Hartel, R.W. 1992. "Solid-liquid equilibrium: crystallization in foods." In *Physical chemistry of foods,* H.G. Schwartzberg and R.W. Hartel (Eds.). Marcel Dekker Inc., New York, 47-81.

Kloek, W.; P. Walstra; and T. Van Vliet 2000. "Crystallization kinetics of fully hydrogenated palm oil in sunflower oil mixtures." *Journal of the American Oil Chemists' Society*, 77, 389-398.

Lambrigger, M. 1996. "Evaluation of data on the kinetics of isothermal polymer crystallization." *Polymer Engineering and Science*, 36(1), 98-99.

Metin, S. and R.W. Hartel 1998. "Thermal analysis of isothermal crystallization kinetics in blends of cocoa butter with milk fat or milk fat fractions." *Journal of the American Oil Chemists' Society*, 75, 1617-1624.

Vanrolleghem, P.A. and D. Dochain 1998. "Bioprocess model identification". In *Advanced instrumentation, data interpretation and control of biotechnological processes*, J. Van Impe; P.A. Vanrolleghem; and D. Iserentant (Eds.). Kluwer Academic Publishers, Dordrecht, 251-318.

Zwietering, M.H.; I. Jongenburger; F.M. Rombouts; and K. Van 't Riet 1990. "Modeling of the bacterial growth curve." *Applied and Environmental Microbiology*, 56, 1875-1881.

AUTHOR BIOGRAPHY

IMOGEN FOUBERT was born in Duffel, Belgium on November 9th, 1976. She graduated from Ghent University in 1999 with a M.Sc. in Biological Engineering Sciences. From 1999 to 2003 she was Research Assistant of the Fund for Scientific Research – Flanders (Belgium) (F.W.O.-Vlaanderen). She obtained her PhD in May 2003. The title of her PhD research was 'Modelling isothermal cocoa butter crystallization: influence of temperature and chemical composition'. It was conducted at the Laboratory of Food Technology and Engineering (Department of Food Technology and Nutrition) in close cooperation with BIOMATH. Currently she is Postdoctoral Fellow of the Fund for Scientific Research – Flanders (Belgium) (F.W.O. Vlaanderen).
E-mail: Imogen.Foubert@UGent.be

PETER A. VANROLLEGHEM is professor at Ghent University since October 1st 1997. He is responsible for the discipline model-based bioprocess optimization. In addition to his teaching in bioengineering, he has supervised 10 PhD's to completion in addition to 53 MSc's, all of them in Applied Biological Sciences. Currently 10 MSc students are active under his supervision and 22 PhD studies are ongoing (with 3 public defences expected this year). His main research subjects are the monitoring, modelling and optimization of biotechnological processes in agriculture, food and feed industry, and, mainly environmental systems. His research unit has grown linearly with time and so has the publication output. Since the beginning it has maintained an average of one publication per collaborator per year in international journals with peer review in addition to one oral presentation at conferences (with proceeding publication). In total he has published over 120 papers in peer reviewed journals.
E-mail: Peter.Vanrolleghem@UGent.be
Webaddress: http://biomath.ugent.be

KOEN DEWETTINCK was born in Ghent (Belgium) in 1968. He graduated from Ghent University in 1991 as Bio-Engineer in Chemistry. From the same university, he obtained a Ph.D. in Applied Biological Sciences in 1997. Since 1999, he is full-time docent in Food Technology and Engineering. He is head of the Laboratory of Food Technology and Engineering with research areas such as technology and functionality of dairy products and of oils and fats, powder technology, functionality of biopolymers and food authenticity. Since 2003, he is director of the Flemisch Technology Advisory Centre for the chocolate, biscuits and confectionary industries (VLAZ). Since 2004, he is director of the Flemisch Technology Advisory Centre for the vegetable processing industries (VLAG).
E-mail: Koen.Dewettinck@UGent.be
Webaddress: http://www.fte.ugent.be

EXPERIMENTAL AND CFD STUDY OF POWER CONSUMPTION WITH GATE AGITATORS - EFFECTS OF BOTTOM CLEARANCE AND SCALE UP

Guillaume Delaplace[1*], Romuald Guérin[1], Anne Moreau[1], Claude Torrez[2], Christophe André[2] and Jean-Claude Leuliet[1]

[1] Institut National de la Recherche Agronomique (*I.N.R.A*), Food Process Engineering and Food Technology Laboratory, 369 Rue Jules Guesde, B.P. 39, 59651 Villeneuve d'Ascq Cédex, France.

[2] Hautes Etudes Industrielles (*H.E.I.),* Process Engineering Laboratory, 13 Rue de Toul, 59046 Lille, France

* E-mail : delapla@lille.inra.fr

ABSTRACT

In the last few years, the application of computational fluids dynamics (CFD) in stirred vessel gets more and more numerous and CFD software are now having a significant impact in this area. In contrast to this, degree of resolution that can be achieved with numerical results are seldom been examined in detail. For example, even for single fluid flow under laminar regime, the question is still raise whether numerical predictions can detect low variations of global design parameters of the mixing systems. In this study, experimental measurements and CFD simulations were carried out to answer this issue. The global design parameter was power consumption when mixing Newtonian fluids with various gate agitator/vessel arrangements and minor variations of this global parameter was induced by changes of bottom clearance position. In this work, scale up results obtained experimentally and numerically from 3D-CFD simulations (FLUENT 5.4) have also been presented and discussed.

KEY WORDS

Gate agitator, Newtonian , CFD, Power consumption, bottom clearance, scale-up.

INTRODUCTION

Mixing of highly viscous fluids in agitated vessel is one of the oldest and of the most common processing operation encountered in many industrial applications. A large number of mixing systems can be used to achieve mixing operations depending on the mixing duty. Traditionally, the design of agitators has relied upon laboratory testing and generalised correlations of extensive experimental data.

Over the last decades, the application of computational fluids dynamics (CFD) in stirred vessel get more and more numerous and CFD software have achieved considerable progress in the modelling of real flows. Consequently, at the present time, it is widely admitted that CFD software will become in a close future an essential tool to size and design efficiently mixing equipments. In contrast to this, limited scientific papers have been devoted to examine the ability of CFD software to provide accurate and reliable data when optimisation of the stirred tanks requires minor change of geometric parameter of the mixing systems. Indeed, when it is necessary to rebuild a new mesh to compare efficiency of two mixing phenomena caused by two different positions of an agitator, few studies are available to estimate the relative accuracy of numerical predictions.

For that reason, the present work investigates the performances of a CFD software (FLUENT 5.4) to predict variations of mixing phenomena caused by minor change of the geometrical parameters of the stirred tank.

To do that, experimental power input measurements were carried out with highly viscous Newtonian fluids for eight gate agitator/vessel arrangements located at various bottom clearance values. On the basis of these experimental data, it was possible to determine functional dependence of power consumption on bottom clearance for each mixing system. Then experimental data were compared with those obtained by numerical simulation in order to get a valuation of ability of CFD software to predict accurately minor variations of geometric parameters of the mixing systems.

Although it is also recognised that numerical simulation represent an efficient pathway to scale-up or scale down mixing equipments, there are very few reports concerning comparison between experimental and numerical data on scale up or/and scale down phenomena. So, another purpose of this paper is to address this issue. Consequently, some of the gate agitator/vessel arrangement have been scaled up and power consumption when mixing Newtonian fluids have also been measured. Then, numerical data were compared with those obtained experimentally and the agreement between the two set of values was discussed.

EXPERIMENTAL APPARATUS AND METHOD

Mixing system characteristics

For the experimental measurements, eight gate agitators and four flat bottomed vessels with increasing diameter have been used. Picture of the impellers (named *A1*, *A2*, *A3*, *B*, *C*, *D*, *E*, *F*) and of the tank (named *V1*, *V2*,V3,*V4*) are given respectively in Figures 1 and 2. Symbols and geometrical parameters of the mixing systems are respectively reported in Table 1 and Table 2 and are indicated in Figure 3. Trials were carried out with various heights of liquid. However most of the results were obtained for a specified H_L/D ratio close to 1.35.

Bottom clearance ratio expressed as $C_B/(H_L - H)$ ranged from 0.07 to 1.

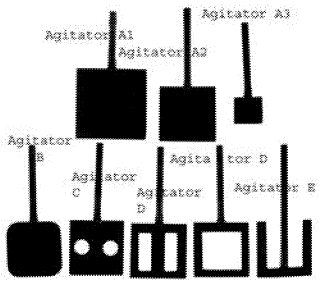


Figure 1 : Picture of the gate agitators investigated in this study.

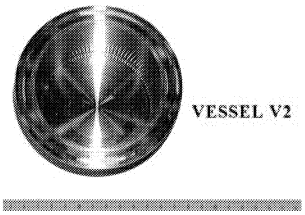


Figure 2 : One example of a flat bottomed vessel used in this study (tank V2).

Table 1 : Dimensions of the gate agitators studied

Agitator	Agitator diameter, D (10^{-3} m)	Agitator height, H (10^{-3} m)
A1	50.8	50.7
A2	40.6	40.6
A3	20.5	19.6
B	39.9	40.6
C	39.0	40.2
D	40.6	40.1
E	39.9	40.2
F	40.1	40.4

Table 2 : Dimensions of the vessel studied

Vessel	Tank diameter, T (10^{-3} m)	Height of the vessel, H_T (10^{-3} m)
V1	105.0	102.0
V2	80.0	60.0
V3	41.9	60.0
V4	53.1	88.0

Fluids

Experiments were carried out with mixtures of glucose syrups with various water percentages at room temperature. The rheological properties and densities of these Newtonian fluids were the following : Viscosity: 2.1 < μ [Pa.s] < 70 and Density: 1360< ρ [kg/m^3] <1415.

Experimental methodology

It is widely recognised that the power input is possibly one of the most important design parameters for a stirred tank. Over the years, a wealth of information has accumulated on the power consumption for a variety of mixing equipments. Most of these articles concerned the role of rheological properties and/or the influence of geometrical parameters of the mixing system (tank+agitator) on power consumption as underlined recently by Delaplace et Leuliet (2000)

From a literature review of investigations on power draw with various mixers, it appears that the main geometric parameter studied is the side wall clearance (i.e the gap between the vessel wall and the rotating impeller). This has been shown to be a significant variable exerting strong influence on power consumption even for the mixing of Newtonian fluids. In contrast to this, power consumption is known to be slightly dependent on bottom clearance as shown by Cheng et *al.* (1995) and by Espinosa Solares et *al.* (1997).

In this work, bottom clearance was chosen as the variable geometric parameter and power consumption was the measured quantity. To do that, a rheometer system with a pneumatic lifting device was used (Figure 3). The rheometer system allowed us to measure both power consumption inferior to 10^{-7} Watt and to locate precisely the impeller from the bottom of the vessel (magnitude of accuracy is higher than 1 µm). On this basis, reliable experimental results were achieved and were available for valuation of CFD predictions.

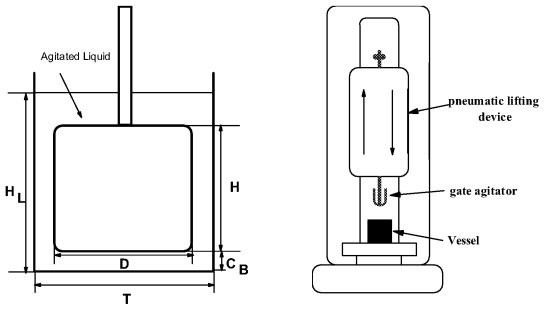


Figure 3 : Geometrical parameters of the mixing system (left) and sketch of the experimental set-up (right).

Numerical simulation

In this work, numerical simulations of the velocity field induced by various agitator/vessel arrangements were also performed. According to nomenclature defined in Table 1, these agitator/vessel arrangements will be named [A1-V1], [A2 -V2], [A3-V3],[F-V2] (Figure 4). Note that mixing systems named [A1-V1] and [A3-V3] are respectively a partly scale down and scale up of agitator/vessel arrangement [A2-V2] as shown in Figure 5. Indeed, all the geometrical parameters of mixing systems [A1-V1] and [A3-V3] excepted the diameter of the shaft can be deduced from agitator/vessel arrangement [A2-V2] applying a fixed scale factor. Diameter of the shaft was kept unchanged to duplicate experimental mixing conditions (for mechanical consideration, the size of the diameter could not be modified).

For each meshed mixing system, the dependence of power input on the ratio C_B/D was studied for at least five bottom clearance values. Note, that for all bottom clearance studied, a new unstructured mesh of tetrahedral cells has been built. For a given agitator/vessel arrangement, the procedure was the following : firstly, mesh nodes have been created along the

edge of the mixing system. Then mesh volume command has been applied to the agitated fluid.
Consequently, cells number of the grid depends on the bottom clearance value, the geometry and size of the mixing equipment. For the meshed mixing system, the number of control volumes of the grid vary between 22,827 and 114,696.
For numerical simulations, the viscosity and the density of the agitated fluid were respectively set equal to 2.3 Pa.s and 1300 kg/m^3. The impeller rotational speed vary between 0.008 and 0.02 rev/s
The numerical tools and solution procedure used by the commercial CFD finite volume software (FLUENT 5.4) to determine the fluid velocity profiles inside the vessel are well known (Pantakar (1980)) and will not be repeated in details here. In the following part, only details about the mixing system modelling options will be briefly summarised:

- the boundary conditions at the impeller shaft and at the vessel wall were those derived assuming the no-slip condition.

-at the free surface, the boundary conditions were modelling by requiring that there is no normal velocity and zero normal gradient for all variables. For the operating conditions tested (μ =2.3 Pa.s, ρ =1300 kg/m^3, 0.008 rev/s < N <0.02 rev/s), the assumption of flat velocity profile was checked experimentally

-a rotating reference frame was used to perform the simulation.

- resolution of the algebraic equations was carried out using the semi-implicit algorithm pressure linked equation (SIMPLE) with a first order upwind discretisation scheme.

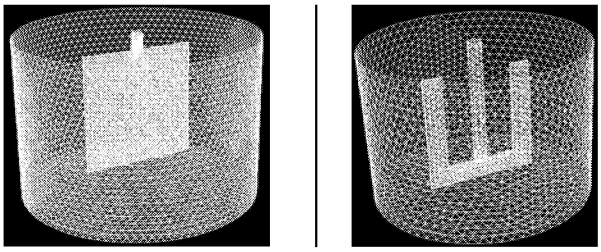


Figure 4 : Examples of impeller/vessel arrangement meshed. Impeller /vessel arrangements [A2-V2] (left) and [F-V2](right).

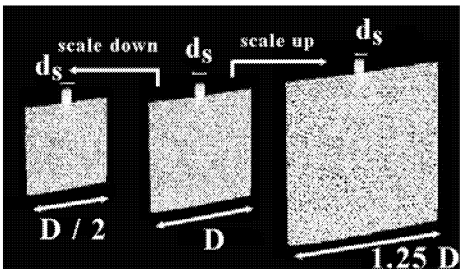


Figure 5 : Partly scale up and scale down of agitator A2 (middle)

Power consumption analysis

For experimental approach, power consumption was deduced from torque and impeller rotational speed measurements. For CFD simulations, power consumption was calculated by two methods. The two methods were compared and sensitiveness to numerical errors for each method was discussed.

The first method assumes that the power consumption supplied by the impeller is exclusively consumed as viscous dissipation of energy inside the vessel and consequently the total power consumption P was calculated by the summation of energy consumed in each volume control, throughout the vessel

$$P = \sum_i \mu \, \dot{\gamma}_i^{\,2} \, V_i \qquad (1)$$

where μ , $\dot{\gamma}_i$, V_i are respectively the Newtonian viscosity, the shear rate and the volume of the agitated fluid contained in cell i . For each cell, $\dot{\gamma}$ was deduced from the second invariant of rate of the deformation tensor $\overline{\overline{\Delta}}:\overline{\overline{\Delta}}$

$$\dot{\gamma} = \sqrt{\frac{1}{2}\overline{\overline{\Delta}}:\overline{\overline{\Delta}}} \qquad (2)$$

The second numerical method allowed to obtain the power consumption from the knowledge of the angular velocity and the computed torque at the surface of the agitator.
Classically, power consumption of a mixing system under laminar regime is described by using the typical relationship

$$N_p \, Re = K_p \qquad (3)$$

where Kp is a constant characteristic of the impeller/vessel configuration.
Consequently, for each mixer , it was chosen to illustrate the effect of bottom clearance on the power consumption by following the evolution of the parameter Kp with the normalized bottom clearance ratio $C_B/(H_L - H)$.

RESULTS AND DISCUSSION

Experimental results with gate agitators

Influence of the impeller/vessel geometry on power input.
Figure 6 showed the typical results obtained experimentally illustrating the Power number-Reynolds number behavior for the gate agitators investigated.
As shown in Figure 6, it was possible for each impeller/vessel configuration studied, to describe the Newtonian power curve obtained in the laminar regime using similar relation as equation (3) and consequently, for each mixer configuration, a unique $K_p\left(= N_p.Re\right)$ value can be deduced. The resulting Kp values are summarised in Table 3. All these experimental values of Kp were obtained with both a fixed liquid height ratio H_L/D close to 1.35 and a impeller diameter D, close to 0.004m.
Analysis of Table 3 allows us to point out that:

- the power input with all the gate agitators are same order of magnitude

- for all the gate agitators studied, values of Kp increase with a decreasing bottom wall clearance. Kp variations are about 25% in the range of bottom clearance ratio considered.

Table 3: Values of power constant *Kp* for various impeller/vessel configurations with 4cm diameter.

	Impeller/vessel arrangment					
$C_{B\,(m)}$	[A2-V2]	[B-V2]	[C-V2]	[D-V2]	[E-V2]	[F-V2]
0.001	-	127.8	-	-	-	142.0
0.002	128.4	122.9	138.0	119.5	132.7	138.6
0.003	126.3	120.0	129.3	117.0	130.7	132.4
0.004	126.5	118.5	124.5	115.4	128.7	131.8
0.005	121.4	114.6	126.3	114.5	122.4	129.9
0.008	118.7	113.2	118.7	109.2	116.4	119.7
0.010	115.2	109.5	114.4	106.9	115.0	120.0
0.012	112.9	109.2	116.1	102.9	113.9	119.5

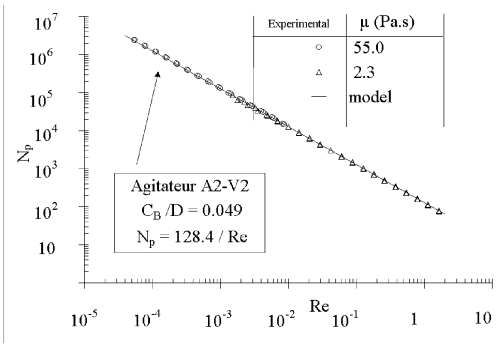


Figure 6 : Power curve under laminar and isothermal regime for one impeller/vessel configuration.

Influence of bottom clearance on power input for the 4 cm diameter gate agitators.

Figure 7 is another reading on the effect of bottom clearance on power consumption, in terms of *Kp* factor. This time evolution of *Kp* factor with ratio $C_B/(H_L-H)$ has been plotted. From analysis of Figure 7, it appears clearly that the effect of bottom clearance on power consumption follow the same trend for each impeller/vessel arrangement. This result was expected. Indeed all test gate agitators are designed on the same basis: there are similar side wall clearance, there are contained in a vertical plane and there are fitted with an horizontal blade at the bottom part of the impeller. Therefore, it is not surprising that for all these agitator the bottom clearance effect is very similar.

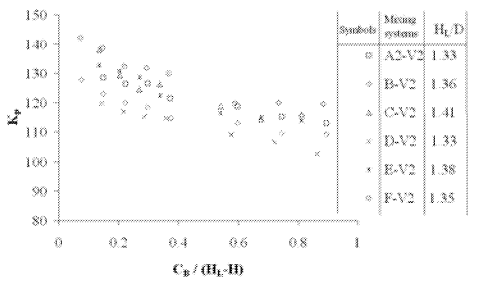


Figure 7 : Effect of bottom clearance on the power constant *Kp* at nearly fixed liquid height ratio H_L/D .Various symbols refer to experimental data.

In order to discriminate the role of bottom clearance on power input between the other geometrical parameters (ie design of gate agitators studied), an attempt has been made to incorporate bottom clearance effect in a correlation independent on the geometry of the gate agitator.

This was done by plotting, for each gate agitator, the evolution of the ratio K_p/K_{pmin} versus $C_B/(H_L-H)$.

For a given impeller/vessel arrangement and a fixed liquid height ratio, K_{pmin} refers to the value of *Kp* factor obtained when the agitator is wholly submerged and just bellow the free surface of the agitated liquid (Figure 8). This minimum value of K_p factor, K_{pmin}, is obtained when the gate agitator is located at a bottom clearance value C_B equal to $C_{Bmax}=H_L-H$ (Figure 8).

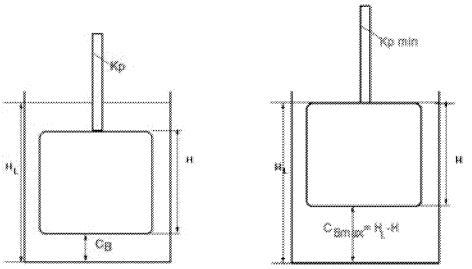


Figure 8 : Definitions of K_{pmin} and C_{Bmax}

As it can be seen from figure 9, for all gate agitators studied, it is possible to describe power input dependence with bottom clearance by a unique curve.

The following equation was found to describe the data generated during the present experimental program:

$$K_p/K_{pmin}=\left(C_B/(H_L-H)\right)^{-0.056} \quad (4)$$

Equation (4) indicates that the ratio K_p/K_{pmin} is proportional to the ratio of bottom clearance value exponent – 0.056.

Weak value of power law exponent confirms that power consumption is very slightly dependent on bottom clearance as found by previous authors (Cheng et *al.* (1995) and by Espinosa Solares et *al.* (1997)). Note that it is clear that equation (4) is strictly applicable over the range of experimental conditions summarised above:

$0.07<C_B/(H_L-H)<1$ and $H_L/D=1.35$

Effect of a scale factor on power input dependence with bottom clearance for gate agitators.

In order to establish the influence of scale factor on equation (4), geometrical scale up and scale down of mixer [A2-V2] have been used. Then experimental evolutions of the ratio K_p/K_{pmin} versus $C_B/(H_L-H)$ for the three mixing systems [A1-V1] [A2-V2] [A3-V3] have been plotted and compared in Figure 10.

For the range of scale factors investigated (scale factors = 0.5 and 1.25), it appears that the effect of scale factor on power input dependence with bottom clearance is not significant.

Consequently equation (4) previously proposed seems to be still valid.

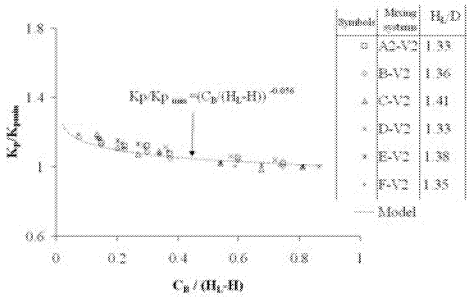


Figure 9 : Effect of the ratio $C_B/(H_L-H)$ on the ratio K_p/K_{pmin} at fixed liquid height ratio H_L/D for various gate agitator/vessel arrangements. Various symbols refer to experimental data.

Effect of liquid height on power input dependence with bottom wall for gate agitators

In order to establish influence of liquid height on power input dependence with bottom clearance, experiments with mixing system [A3-V3] submerged with increasing liquid height have been carried out. Then, for each liquid height, evolutions of the ratio K_p/K_{pmin} versus ratio $C_B/(H_L-H)$ have been measured. Results are shown in Figure 11.

Figure 11 shows clearly that for liquid height ratio encountered ($1.17 < H_L/D < 2.69$) there is no significant scatters between correlation proposed (obtained by power law regression on experimental data when liquid height ratio is equal to H_L/D=1.35) and the other ones (obtained for other liquid height ratio).

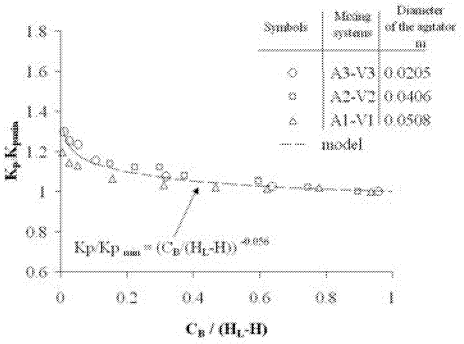


Figure 10 : Effect of scale factor on power input dependence with bottom clearance for gate agitators. Various symbols refer to experimental data

Effect of clearance wall on power input dependence with bottom clearance.

In order to establish the influence of clearance wall on equation (4), power input measurement with increasing wall clearance mixing systems have been carried out. For all the mixing systems tested, the liquid height ratio H_L/D was maintained constant close to 1.36. Then experimental evolutions of the ratio K_p/K_{pmin} versus $C_B/(H_L-H)$ for the various gate agitators/vessel arrangements [A2-V4] [A2-V2] [A3-V3] [A3-V4] have been plotted and compared in Figure 12.

For the range of side wall clearance investigated, C_w/D ($0.1 < C_w/D = 1/2\,(T/D-1) < 0.8$), it appears that the effect of side wall clearance on power input dependence with bottom clearance is not significant. Consequently equation (4) previously proposed seems to be still valid.

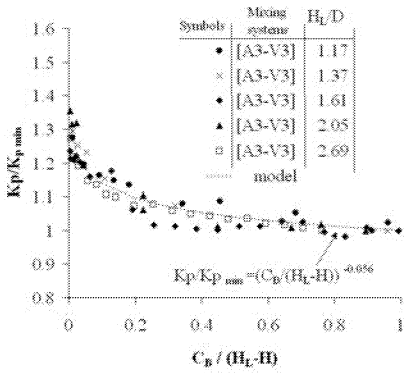


Figure 11 : Effect of liquid height on power input dependence with bottom clearance for gate agitators. Various symbols refer to experimental data.

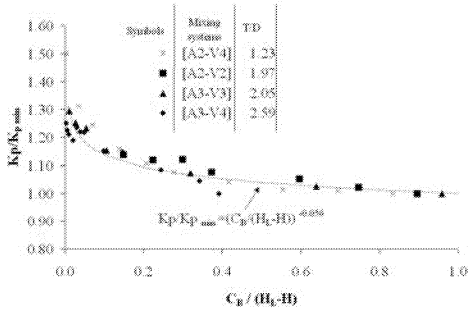


Figure 12 : Effect of wall clearance on power input dependence with bottom clearance. Various symbols refer to experimental data

To sum up experimental data above allowed us to suggest that both scale factor, liquid height ratio ($1.17 < H_L/D < 2.69$) and side wall clearance ($0.1 < C_w/D < 0.8$) have little effects on the power input dependence with bottom clearance. This indicates that, although equations (4) should be regarded in the light of the relatively small diameter of the agitator used in its derivation, these correlations seem to be still valid for a large range of operating conditions and gate agitator/vessel arrangements.

Combining equation (4) for two various bottom clearance values (C_{B1} and C_{B2}) and a fixed liquid height ratio another reading of equation (4) is :

$$K_{p1}/K_{p2} = (C_{B1}/C_{B2})^{-0.056} \qquad (5)$$

where K_{p1} and K_{p2} are respectively the power consumption of the gate agitator/vessel arrangement with bottom clearance values are equal to C_{B1} and to C_{B2}.

It therefore seems advantageous to use equation (5) and the knowledge of mixer power input (in terms of K_p factor) for a fixed bottom clearance value to predict the variation of power consumption due to change in bottom clearance position.

Numerical results with gate agitators

Influence of bottom clearance on power consumption obtained by CFD.

Example of power consumption data obtained using CFD software package (FLUENT 5.4) are reported in Figures 13 and 14.

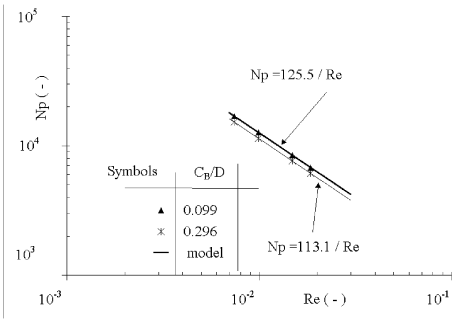


Figure 13 : Power curve obtained numerically under laminar regime and isothermal condition with Newtonian fluids (μ =2.3 Pa.s) for two bottom clearance positions of the gate agitator [A2-V2] (C_B/D =0.099 and 0.296; H_L/D = 1.33). Numerical data reported are obtained by method 1 (see Material and methods)

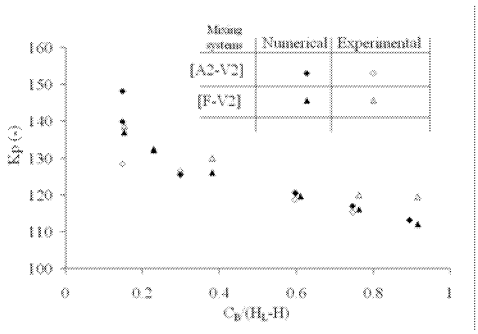


Figure 14 : Effect of bottom clearance on the power constant K_p obtained numerically for two gate agitators of 4cm Diameter([A2-V2]; [F-V2]) with a fixed liquid height ratio $H_L/D = 1.35$. Numerical data reported are obtained by method 1 (see Material and methods)

Note that for these figures, power consumption values are based on the viscous dissipation but similar results are obtained by the other numerical method used to predict power consumption (based on torque at the surface of the agitator). This will be discussed in detail latter.

Figure 13 shows clearly that, as expected, the Newtonian power curve obtained in the laminar regime by CFD method can be described by traditional relationships $N_p\ Re = K_p$.

Moreover, as it can be seen from Figure 14, for the two 4 cm diameter gate agitators studied, the numerical values are in good agreement with the experimental data.

Effect of a scale factor variation on power consumption dependence with bottom clearance

As shown in Figure 15, for the range of scale factors investigated (scale factors = 0.5 and 1.25), it appears that the numerical values obtained by the first numerical method tested (based on viscous dissipation) agree reasonably well with the experimental points.

Nevertheless, it seems that magnitude of scatters between experimental and numerical values are not constant but dependent on bottom clearance values. Indeed, for the gate agitators tested highest relative errors are obtained for lowest bottom clearance values. This may be explained by the way used to calculate power consumption. Indeed the power consumption based on the numerical viscous dissipation is very sensitive to the quality of the mesh (Skewness of control volumes) and for low clearance bottom values it is very difficult to obtain good quality of mesh between the bottom part of the agitator and the vessel if number of cells kept unchanged. Consequently accuracy is poor.

On the contrary, a calculation based on determination of torque at the surface of the agitator gives better results in this case since not dependent on bottom clearance values. This clearly appears in Figure 16.

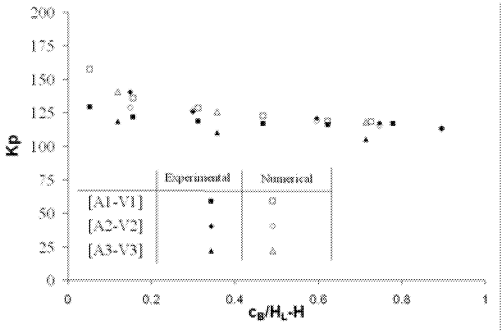


Figure 15 : Effect of scale factor variation on power input dependence with clearance wall. Numerical data reported are obtained by method 1 (see Material and methods)

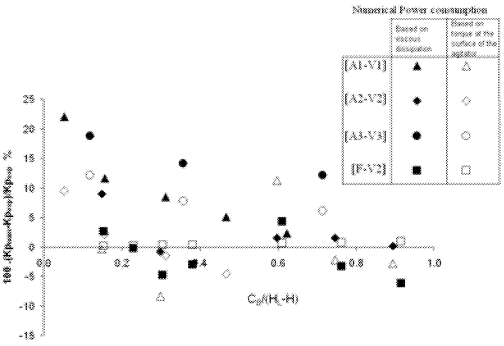


Figure 16 : Relative error obtained numerically on power input for various gate agitators. Numerical data reported are obtained by method 1 (Viscous dissipation) and method 2 (torque at the surface of the agitator)

Numerical results plotted in Figure 15 also show that, as it was found for experimental approach, the effect of scale factor on power input dependence with bottom clearance is not significant. Indeed all numerical power data can be described by a unique curve which is quite logical since mixing systems [A1-V1] and [A3-V3] are quasi-exact geometrical extrapolations of impeller/vessel arrangement [A2-V2] (except

the diameter of the shaft). This proves in a certain extent that numerical method is well adapted and accurate enough to predict scale up phenomena.

In this paper, we have investigated the effect of scale factor variations on power input dependence with bottom clearance in a relatively small range of scale factors. Then it can be expected that changing more strongly the scale factor may produce different results. Although it was not possible to validate the numerical results by experimental work, we have decided to show you numerical power consumption predicted for agitator [A2-V2] when scale factor has been multiply by 10. Results are plotted in Figure 17.
Analysis of figure 17 shows, even for important scale factor, power input dependence with bottom clearance obtained by numerical approach seems not differ from that measured experimentally with small mixers. So, assuming this result is good, we can suppose that scales up for power consumption rules will be achieved in this case.
Note that this time there is less discrepancy between the set of numerical values obtained for mixing system [F-V2] and its scale up compared to previous results obtained with gate agitator [A2-V2]. This may be explained by the fact, this time all the geometrical parameters of mixing system have been exactly extrapolated without approximation about the size of the shaft diameter

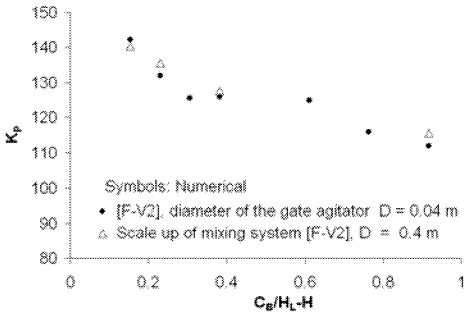


Figure 17 : Effect of scale factor variation on power input dependence with clearance wall for anchor agitator. The scale factor between the two mixing system is equal to 10. Numerical data reported are obtained by method 1 (see Material and methods)

CONCLUSION

On the basis of this numerical and experimental study, the effect of bottom clearance on power consumption with various gate agitators using Newtonian fluids has been investigated.
Experimental measurements have allowed us to suggest a unique correlation to predict power consumption dependence with bottom clearance values for gate agitators. Moreover, this correlation was shown to be valid for various scale factors, liquid heights and side wall clearances
Although a new mesh is required for each bottom clearance position, numerical results confirm fairly well the experimental points and show that CFD methods are accurate enough to predict weak variations of global design parameters (Power consumption) caused by minor change of geometrical parameters (bottom clearance) of the mixing systems.

Finally, scale up and scale down results obtained experimentally have also been compared to numerical data. It has been shown that for range of factor scale investigated the two set of data agree reasonably well which proved that numerical approach is also able to give satisfactory trends for scale up.

NOMENCLATURE

C_B bottom clearance (m)
C_W side wall clearance (m)
D diameter of the agitator (m)
H_L total liquid depth (m)
H agitator heigh (perpendicular to shaft) (m)
N agitator speed (Rev.s^{-1})
P power consumption (W)
T tank diameter (inside) (m)

Greek Letters

$\dot{\gamma}$ shear rate (s^{-1})
μ Newtonian viscosity (Pa.s)
ρ density (kg/m^3)

Dimensionless number

Kp Power constant defined by Equation (3), (-)
Np Power number, $Np=P/\rho N^3 D^5$, (-)
Re Reynolds number for Newtonian fluids, $Re=\rho N D^2/\mu$, (-)

REFERENCE

Cheng J. , P. J. Carreau and R.P.Chhabra. 1995. "On the effect of wall and bottom clearance on mixing of viscoelastic fluids", Industrial Mixing Fundamentals with Aplications, *AIChE Symposium Series* N°305, 91, 115-122.

Delaplace G. and J.C. Leuliet. 2000. "Power Consumption of Helical Ribbon Impellers in Highly Viscous Liquids – A Review" , *Entropie,* 227, 10-21.

Espinosa-Solares T., E. Brito-de La Fuente , F.L. Thibault and P.A. Tanguy. 1997. "Power consumption with anchor mixers: Effect of bottom clearance", *Chem. Eng. Comm.*, 157, 65-71.

Patankar S.V. 1980. "*Numerical heat transfer and fluid flow*" Series in Computational Methods in Mechanics and Thermal Sciences, Mc Graw Hill, New york, N.Y.

Acknowledgments

The authors are grateful to E. VIDAL for his valuable work in designing and obtaining the experimental set up presented here.

AIRCRAFT PERFORMANCE SIMULATION UTILIZING THRUST VECTORING

Prof. Assim. H. Yousif
Department of Machinery Engineering
University of Technology
Baghdad, Iraq, P.O. Box:35010

Dr. Arif A. Al-Qassar
Al-Rasheed College of Engineering
University of Technology
Baghdad, Iraq, P.O. Box: 35010
E-mail: arifqassar@yahoo.com

Dr. Wisam A. F. Mohammed
Department of Technical Education
University of Technology
Baghdad,Iraq,P.O.Box:35010

KEYWORDS
Enhancement, Performance, Aircraft, Supersonic

ABSTRACT

The aircraft maneuverability enhancement may be achieved by applying a thrust vectoring technique. A three dimensional modified low order panel method supported by semi-empirical skin friction and base drag models are used to predict the aerodynamic coefficients of the aircraft. Fortran computer program (PTVR-1) has been prepared to simulate the flying aircraft. This program was tested to satisfy the output acceptance. The program output shows a good agreement with the published data obtained from wind tunnel and flight tests.

The specific excess power contours for chosen F-15 aircraft obtained in the presence of thrust vectoring are improved in subsonic regions especially at load factor ranging between 1 to 6, whereas there is a slight decrease in the supersonic regions. Incorporation of thrust vectoring provides an enhancement in instantaneous maneuvering by increasing the turn rate with minimum turn radii.

INTRODUCTION

However, the aircraft aerodynamic coefficients play a substantial role in predicting the aircraft performance. These aerodynamic coefficients are obtained by computational methods, computational methods provide the required capability for accurately modeling the aerodynamic interference effects of arbitrary and complicated geometry such as multiple lifting surface and/or wing body configuration. The surface singularity (i.e. panel method) has been developed and extensively applied to the aerodynamic design of subsonic and supersonic flying bodies (Bengellink 1988).

Aircraft thrust vectoring is an innovative technology that has been studied for a number of years. It becomes a big issue in the 1980's and early 1990's with the majority of aerospace companies pushing ahead with related programs. Simply put, thrust vectoring is a change in the direction of thrust produced by an aircraft's engine (Kowal 2002). There are two basic types of TV, 2D and 3D. Two dimensional vectoring (2DTV) works by directing the exhaust up or down (pitch vectoring), While three dimensional vectoring (3DTV) adds the ability to direct the thrust left to right (yaw vectoring). To satisfy the trimming conditions, a control surface (controllable canard) must be added in front of the main wing of the aircraft (Arif 2004).

Details of the selected supersonic aircraft configuration (F-15), and the main characteristics and performance data of the turbofan engine Pratt and Whitney F-100-PW-100 has been obtained from reference (Nicolai 1975). This aircraft was used to test the program. The block diagram in Fig. 1 shows the aircraft performance categories treated in the present paper.

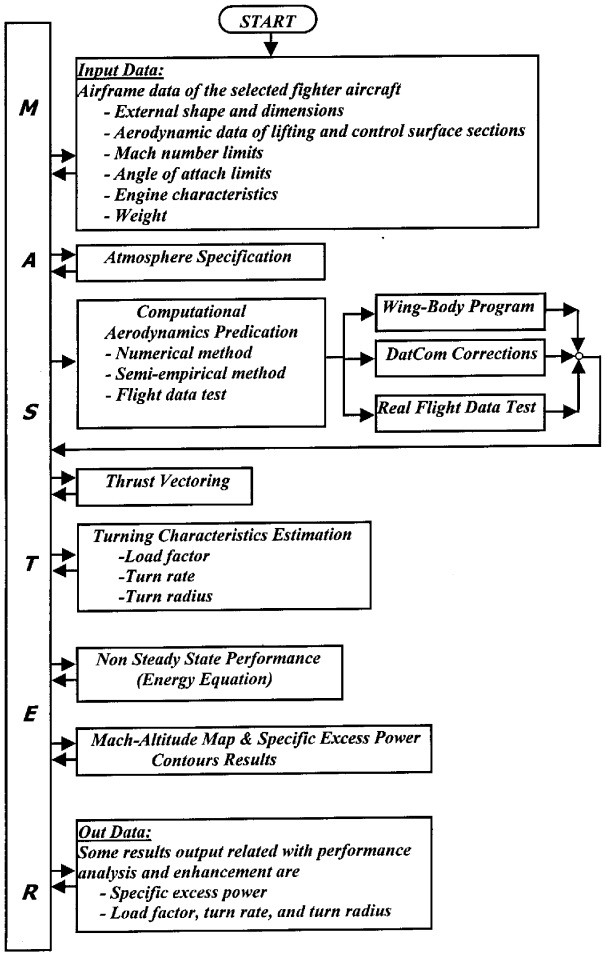


Fig. 1: Block Diagram of "PTVR" Code

COMPUTAIONAL AERODYNAMICS COEFFICINTS

1. Aerodynamic Formulation

The solution is based on the linearized, three-dimensional potential equation (Fornasier 1984).

$$\beta^2 \phi_{xx} + \phi_{yy} + \phi_{zz} = 0 \tag{1}$$

where β^2 is the compressibility parameter and ϕ is the potential function of the perturbation velocity.

Equation (1) is valid for both subsonic and supersonic flows. It should be noted here that the coordinate system used for aerodynamic analysis follows the x-axis in the direction of the free stream flow and y-axis completing the right-hand system.

The fundamental solution of the potential flow equation (1) is used to solve the problem of computing pressure, forces, and moments of the wing-body-tail configurations. The configuration surface is divided into a large number of panels as shown in Fig. 2, each of which contains an aerodynamic singularity distribution.

The boundary conditions of tangential flow are applied on the analogous body, and the resulting incompressible perturbation velocities are transformed back to the real body by:

$$\begin{Bmatrix} u \\ v \\ w \end{Bmatrix} = \begin{bmatrix} 1/(1-M_\infty^2) & 0 & 0 \\ 0 & 1/\sqrt{1-M_\infty^2} & 0 \\ 0 & 0 & 1/\sqrt{1-M_\infty^2} \end{bmatrix} \begin{Bmatrix} u_a \\ v_a \\ w_a \end{Bmatrix} \tag{2}$$

where u, v and w are axial, lateral, and normal velocities perturbation respectively.

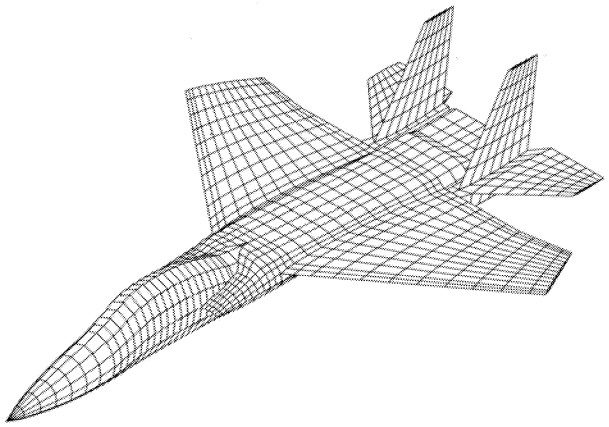


Fig. 2:Three Dimensional Paneling Model of the Aircraft

2. Formulation of the Linear Equations

A system of linear equations is established to relate the magnitude of the velocity normal to the aerodynamic singularity strength. To evaluate the coefficients of this system of equations, the geometrical relationship between each influencing panel and control point is required. A reference coordinate system is used with the origin at the tip of the fuselage nose assuming the aircraft is symmetric about xz-plane.

The resultant velocity normal to panel (i) at the control point is the sum of the normal component of the free stream velocity vector and the normal velocities induced by the panel singularity distributions. Assuming unit magnitude of the free stream velocity vector, and laying in the xy-plane at angle (α) to the x-axis ,then the component of the velocity vector normal to panel (i) is given by (Woodward 1973):

$$w_i = \sin(\alpha)\cos(\theta_i)\cos(\delta_i) - \cos(\alpha)\sin(\delta_i) \tag{3}$$

where θ and δ are panel inclination angles

The resultant normal velocity at the control point (i) is given by:

$$n_i = n_j + \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij} \gamma_j \tag{4}$$

where a_{ij} is the aerodynamic influence coefficient and γ_j is the strength of the j^{th} singularity.

The normal components of the velocity induced at specified control points by each singularity distribution are calculated and make up the coefficients of the system of linear equations relating the strengths of the singularities to the magnitude of the normal velocities.

The boundary condition equations are optioned by setting $\{n_i = 0\}$ in equation (4), the complete set equations may be written in matrix form:

$$[A_{ij}]\{\gamma_j\} = -\{w_i\} \tag{5}$$

where A_{ij} is the matrix of aerodynamic influence coefficients.

In general, the matrix is subdivided into four partitions in order to simplify the calculation and the final solution：

$$\begin{bmatrix} [ABB] & [ABW] \\ [AWB] & [AWW] \end{bmatrix} \begin{Bmatrix} \{\gamma_B\} \\ \{\gamma_W\} \end{Bmatrix} = -\begin{Bmatrix} \{w_B\} \\ \{w_W\} \end{Bmatrix} \tag{6}$$

where B and W are abbreviations of the aircraft body and wing respectively.

The singularity strength which satisfies the boundary condition of tangential flow at the control points, for a given Mach number (M) and angle of attack (AoA), is determined by solving this system of equations.

3. Pressure Load and Aerodynamic Coefficients

Once the strengths of the aerodynamic singularities have been determined, the three components of the velocity at a point (i) will be:

$$u_i = \Delta u_i + \cos AoA \tag{7}$$

$$v_i = \Delta v_i \tag{8}$$

$$w_i = \Delta w_i + \sin AoA \tag{9}$$

The pressure coefficient is calculated using the exact isentropic formula (Zucrow 1976).

$$C_{p_i} = \frac{-2}{\gamma M^2} \left\{ \left[1 + \frac{\gamma - 1}{2} M^2 \left(1 - q_i^2\right) \right]^{3.5} - 1 \right\} \tag{10}$$

where $q_i^2 = u_i^2 + v_i^2 + w_i^2$

The forces and moments acting on the configuration can then be calculated by numerical integration. The normal force (N), tangential force (T), and pitching moment (M) about the origin of coordinates of panel (i) are given by:

$$N_i = -A_i Cp_i \cos\theta_i \cos\delta_i \quad (11)$$

$$T_i = A_i Cp_i \sin\delta_i \quad (12)$$

$$M_i = N_i x_i - T_i z_i \quad (13)$$

where x_i, z_i are the coordinates of the panel control point

The total force and moment coefficients acting on the configuration are obtained by summing the panel forces and moments on both sides of the plane of the symmetry.

$$C_N = \frac{1}{S}\sum_{i=1}^{N} 2 * N_i \quad (14)$$

$$C_T = \frac{1}{S}\sum_{i=1}^{N} 2 * T_i \quad (15)$$

$$C_M = \frac{1}{S\overline{C}}\sum_{i=1}^{N} 2 * M_i \quad (16)$$

Finally, the lift and drag coefficients are:

$$C_L = C_N \cos AoA - C_T \sin AoA \quad (17)$$

$$C_D = C_N \sin AoA + C_T \cos AoA \quad (18)$$

4. Skin Friction Drag Modeling and Base Pressure Estimation

The total or average coefficient of the skin friction drag of the surface is found by integrating the local values along the length of the surface considered, the skin friction drag coefficient is calculated using the method given from (Jankovic 1979).

The base of the body experiences a large negative pressure (relative to ambient or free stream static pressure) resulting in an increase in body drag (Francis 198). The base drag coefficient of the body C_{D_b} is related to the base pressure coefficient as follows:

$$C_{D_b} = -C_{P_b}\frac{S_b}{S} \quad (19)$$

where: S is the wing surface area, C_{P_b} is the base pressure coefficient obtained from reference (Jankovic 1979) as a function of Mach number $C_{P_b} = f(M)$ and S_b is the base area.

ENERGY CONCEPT

For supersonic high-performance aircraft the conventional approach is that used in performance analysis is not adequate for determining best or optimal performance, because the altitude and velocity change instantaneously. The nonlinear differential equations were numerically solved to analyse the non steady state maneuvering thus obtaining the optimal trajectory.

The total energy (E) possessed by an aircraft at any instant comprises its potential energy (due to altitude), its kinetic energy (due to airspeed), and its rotational energy (due to pull-ups and rolls), neglecting the rotational energy by considering the aircraft to be a point mass.

$$E = W(h + \frac{V^2}{2g}) \quad (20)$$

where: W is the aircraft weight, h is the altitude and V is the flight speed.

The specific energy is defined as the energy height (Garret and Watt 1989), expressed as.

$$E_S = \frac{E}{W} = h + \frac{V^2}{2g} \quad (21)$$

The rate of change of the energy height (specific excess power SEP), at the specified speed and altitude can be obtained as follows:

$$SEP = \frac{dE_S}{dt} = \frac{dh}{dt} + \frac{V}{g}\frac{dV}{dt} \quad (22)$$

From Equation (22) the specific power is simply the sum of the rate of climb and the acceleration of the aircraft along the flight path, and can be related to the aircraft characteristics and the flight parameters through the forces equation in the direction of flight, which is

$$T - D - W\sin\gamma = \frac{W}{g}\frac{dV}{dt} \quad (23)$$

where: T is the engine thrust, D is the thrust required and γ is the flight path angle.

However, $V\sin\gamma$ is the rate of climb, so that

$$SEP = \frac{dE_S}{dt} = \frac{dh}{dt} + \frac{V}{g}\frac{dV}{dt} = \frac{(T-D)V}{W} \quad (24)$$

This equation for energy rate, SEP, is extremely useful for determining the ability of the aircraft to change its energy state.

The specific excess power contours at different g-load flight with maximum afterburner power setting, at various points throughout the operating envelope were predicted and plotted against the altitude and Mach number on the specific energy plots to display the total performance capability of the selected supersonic aircraft (F-15) with and without using the thrust vectoring.

Figures 3 to 6 depict the SEP contours for g's= 1, 3, 6 and 9 respectively with and without using the TV . It is clear from these figures that the SEP contours with using TV are improved in subsonic regions. These improvement were strongly appears at g's= 1, 3, and 6, In supersonic regions, the use of TV show no benefit in SEP and that may be due to increase in the induced drag which yield to increase the thrust required. All information needs to study these figures about the AoA degrees and TV angle (δ_N) was attached in these figures.

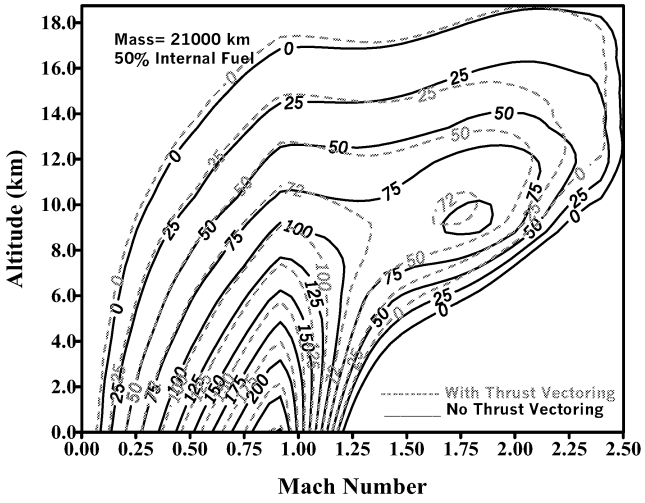


Fig. 3: *SEP* Contours Comparison of Aircraft with and without *TV* at g's= 1.0, δ_N = 20 deg

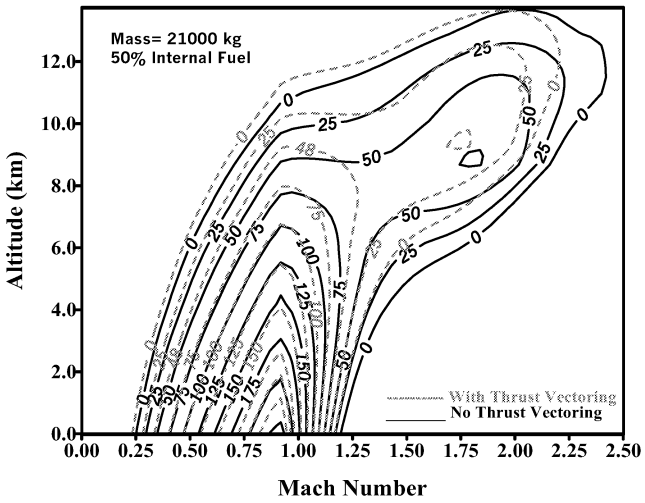


Fig. 4: *SEP* Contours Comparison of Aircraft with and without *TV* at g's= 3.0, δ_N = 20 deg

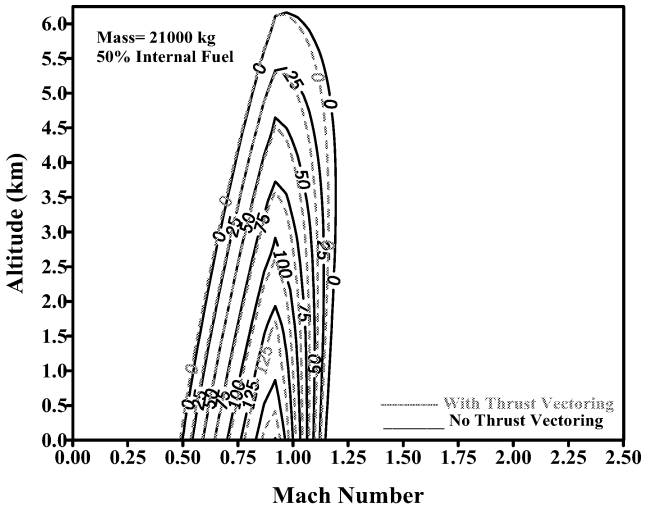


Fig. 5: *SEP* Contours Comparison of Aircraft with and without *TV* at g's= 6.0, δ_N = 20 deg

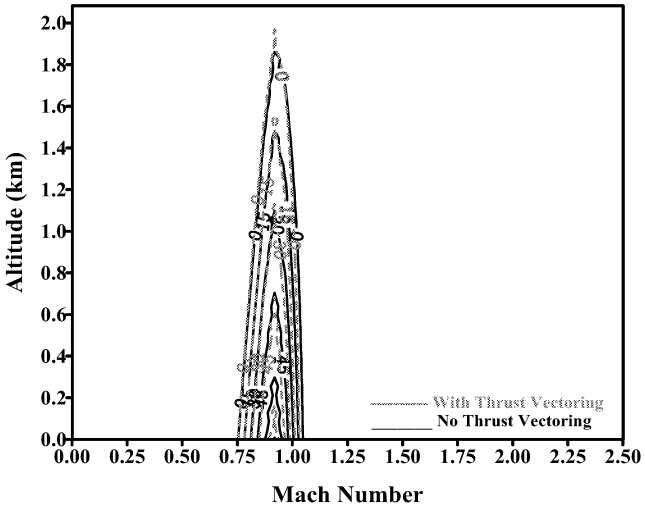


Fig. 6: *SEP* Contours Comparison of Aircraft with and without *TV* at g's= 9.0, δ_N = 20 deg

TURNING PERFORMANCE

The curvilinear flight of an aircraft in the horizontal plane during which the flight path is turned through 360 deg is called horizontal turn (Barnard and Philpott 1989). Explicit formulas are used to determine the main important flight parameters of a turning flight maneuvers, such as airspeed, bank angle, load factor, rate of turn, and radius of curvature.

The load factor (n) in term of aircraft design parameters will be:

$$n = \frac{\rho\, C_{Do}\, E_m}{(W/S)}\, V^2 \sqrt{\frac{2\left(T/W - \sin\gamma\right)(W/S)}{\rho\, C_{Do}}\,\frac{1}{V^2} - 1} \qquad (25)$$

where C_{Do} is the zero lift drag coefficient and E_m is the maximum aerodynamic coefficient

The turn rate and the radius of curvature are obtained as follows respectively:

$$\frac{d\psi}{dx} = \frac{g\sqrt{n^2 - 1}}{V} \qquad (26)$$

$$R_{TURN} = \frac{V^2}{g\sqrt{n^2 - 1}} \qquad (27)$$

The maximum turning capability of the aircraft can be represented by the radius and rate of turn. The turn radius and turn rate are predicted as a function of the Mach number and bank angle, with and without using thrust vectoring as shown in Figures 7, 8 and 9. It is noted from these Figures that with increasing the bank angle at constant Mach number, the turn radius decreases and the turn rate increases.

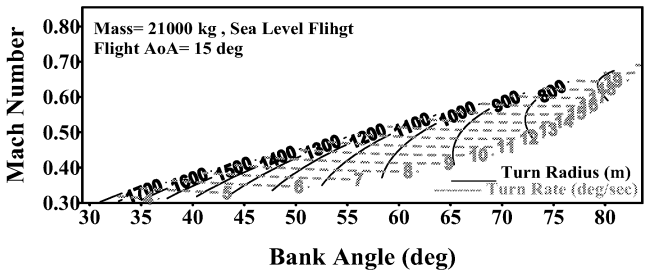


Fig. 7: Turn Radius and Turn Rate as a Function of Mach Number and Bank Angle of Flight AoA= 15 deg

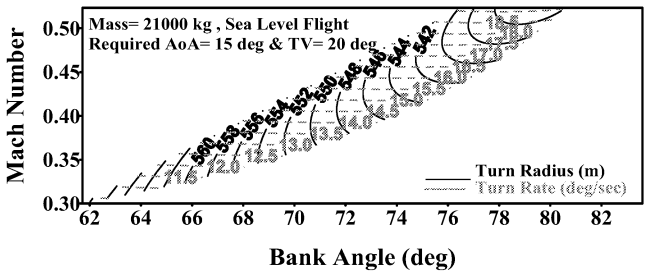


Fig. 8: Turn Radius and Turn Rate as a Function of Mach Number and Bank Angle of Required AoA= 15 deg

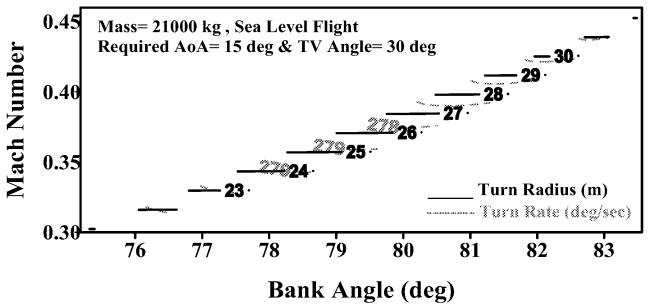


Fig. 9: Turn Radius and Turn Rate as a Function of Mach Number and Bank Angle of Required AoA= 30 deg

CONCLUSIONS

1. An acceptance successful has been achieved to simulate the flying aircraft and to get the aircraft specific excess power and maneuverability. The results and the analysis show by building the (PTVR-1) program a good agreement with data obtained from wind tunnel and flight tests.
2. The specific excess power in presence of thrust vectoring is improved in subsonic regions only, while it not has benefit in supersonic region.
3. Incorporation of thrust vectoring provides an enhancement in instantaneous maneuvring by increasing the turn rate and decreasing the turn radius.

REFERNCES

Anderson J. D., Jr. 1978. *Introduction to Flight*, McGraw-Hill Book Company New York.

Arif A. Al-Qassar. 2004. "Enhancement of Aircraft Performance Utilizing Thrust Vectoring and Reversing", Ph.D. Thesis, Al-Rasheed College of Engineering, University of Technology, Baghdad.

Barnard R. H.; and Philpott D. R. 1989. *Aircraft Flight*, Longman.

Bengellink, R.L., 1988. "The Integration of Computational Fluid Dynamic and Experimental", *AIAA paper*, 88-2043.

Fornasis F. 1984. "Treatment of Supersonic Configurations by an Updated Low-Order Panel Method". *Journal of Aircraft* Vol. 21, No. 4 (Apr), 227-237.

Francis J. H. 1984. *Introduction to Aircraft Performance, Selection and Designs*, John Wiley & Sons Inc., New York.

Jankovic S. 1979. *Aerodinamika Projektila*, Masinsk Fakultet, Univerziteta U Beogradu. Nicolai L. M. 1975. "Fundamental of Aircraft Design", *E. P. Domcone Printing Services*, U. S. A.

Schneider L. G. and Watt G. W. 1989 "Minimum-time thrust Vectored thrust" Journal of Guidance, Vol. 12, No. 6, (Nov-Dec), 777-782.

Woodward F. A. 1973. "An Improved Method for the Aerodynamic Analysis of Wing-Body-Tail Configuration in Subsonic and Supersonic Flow", Part I - Theory and Application, (May).

Zucrow M. J.; and Hoffman J. D. 1976. *Gas Dynamics*, Volume 1, John Wiley & Sons Inc., New York.

AUTHER BIOGRAPHY

ASSIM H. YOUSIF joined the Department of Mechanical Engineering at University of Technology in 1984 as a time lecturer. In September 1999 he became professor of turbomachinary. His main research interest is in the field of aerodynamics. This includes turbomachinary, aeroengine, aircraft and helicopter: Professor Yousif has more than forty published papers and texts books; he has been involved in design and construction of aerodynamics labs. He supervised more than forty postgraduate students.

ARIF A. AL-QASSAR was born in Mosul, Iraq and went to the Military Technical College; he obtained his degree in 1986. He worked for two years for the MTC before moving in 1988 to ENSMA in France to obtain Ph. D, but he returned to Iraq in 1990 because the Golf war. He obtained his M.Sc and Ph.D in 1995 and 2004 from the University of Technology, where he is still working.

E-mail: arifqassar@yahoo.com

A TRAINING ALGORITHM FOR PARTIAL RECURRENT NEURAL NETWORKS

Muaiad Shawqi Murad Al-Faysale (Ph.D.)
University of Technology
Baghdad
Iraq
Email: malfaysale@yahoo.com

KEYWORDS

Neural Networks, Recurrent Networks, Partial Recurrent Networks, Elman Network, Identification.

ABSTRACT

In this paper, a method is for training Partial Recurrent Neural Networks (PRNNs) that makes their structure more powerful is presented. This is achieved by using two adaptable feedback connections. The new training algorithm can be useful in solving many problems in signal processing, control and identification. Simulation results obtained from the identification process of linear and nonlinear systems shows that this algorithm gives better response from that of Multilayer Feedforward Neural Network (MLFFNN) and Modified Elman Network (MELN). This is in term of minimizing the Root Mean Square (RMS) error and training cycles.

INTRODUCTION

Neural Networks (NNs) have become important tools in many applications. Structurally, they can be divided into two types, feedforward and recurrent. Although the structure of the former one is simple, it has no dynamic capability. While the later one has dynamical structure, but it is difficult in analysis. The topology of recurrent networks is shown in figure (1).

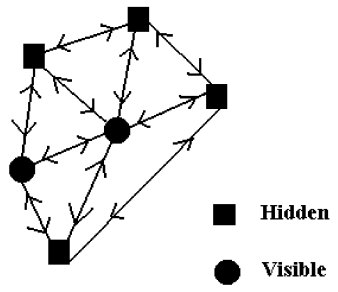


Figure. 1. Recurrent Networks.

Recurrent Neural Networks (RNNs) are networks with feedback loops in their inter-node connection paths (Kanjilal 1995). They can be further divided into two types, depending upon the connections between layers as fully and partially recurrent network (Kwok et al. 1994). Partially Recurrent Neural Networks (PRNNs) contain the advantages of MLFFNN and RNN. PRNs such as Jordan network and Modified Elman Network (MELN) (Elman 1990a; Elman 1991b) extended the MLFFNN with context units having processing elements that remember past activity. Context units are required when patterns are over time, i.e. the past values of the network influences the present processing. Many researchers paid a lot of attention to PRNN by applying it in signal processing (Santos and Duro 1998), network generalization (Kothari and Agyepong 1997), control (Chen and Khalil 1992) and identification (Sastry et al. 1994; Kwok et al. 1994).

A NEW STRUCTURE FOR RNN

The MELN with fixed feedback gain of the feedback-connection, and have the same value for all self-connections (Pham and Liu 1997), can be modified in the following way. The input to the context units, as shown in figure (2), is a collection of two vectors of adaptable weights. One of them is multiplied by the previous output of the hidden units while the other is multiplied by the previous output of the context units. Therefore, the present output of the c^{th} context unit at time k is given by:

$$M_c^{co}(k) = f[\beta c(k)\, M_c^{co}(k-1) + \zeta_c(k) H_j(k-1)] \quad (1)$$

$$M_c^{co}(k) = f[\zeta_c(k) H_j(k-1) + \beta_c(k)\, \zeta_c(k-1)\, H_j(k-2) + \beta_c(k-1)\, \zeta_c(k-2)\, H_j(k-3) + \ldots.] \quad (2)$$

Where (c=1,2…C) is the number of the context units, (j=1,2…J) is the number of the hidden units, β_c(Cx1) and ζ_c (Cx1) are the adaptable weights, H_j(k-1) and M_c^{co}(k-1) are the past outputs of the hidden and context layer and f(.) is a linear or nonlinear activation function.

The algorithm presented in this paper differ from the one proposed by (Kwok 1994). In this algorithm, the self-feedback (β) has an adaptable value and for each context

unit, meanwhile the algorithm proposed by (Kwok 1994) has one adaptable value for all context units. Moreover, the feedback connection gains, between the hidden and context units (ζ), are adaptable and have a value for each connection, while for the one proposed by (Kwok 1994), they have the same fixed value.

The proposed algorithm is called Modified Recurrent Network (MRN) and its structure is shown in figure (2).

For the MRN, the following equations are hold:

$$H_j(k)=f\left[\sum_{i=1}^{I} W_{jc}^{hc} M_c^{co}(k)+\sum_{i=1}^{I} W_{ji}^{hu} u_i(k)\right] \quad (3)$$

$$M_c^{co}(k) = f\left[\beta_c(k) M_c^{co}(k-1) + \zeta_c(k) H_j(k-1)\right] \quad (4)$$

$$O_l(k) = f\left[V_{lj} H_j(k)\right] \quad (5)$$

Where W_{jc}^{hc}, W_{ji}^{hu} and V_{lj} are the weights matrices, *f(.)* is a linear or nonlinear activation function, *O=1,2,....L* is the network output, and *u* is the input vector. Moreover, *j=1,2...C* is the number of the hidden layer nodes which is equal to the number of context nodes in this work.

The update of the weights in the MRN are given by:

$$\Delta V_{lj}(k) = \eta(T(k) - O_l(k)) f'(O_l(k)) H_j(k) \quad (6)$$

$$\Delta W_{jc}^{hc} = \eta(T(k) - O_l(k)) f'(O_l(k)) V_{lj} f'(H_j(k)) \frac{\partial H_j(k)}{\partial W_{jc}^{hc}} \quad (7)$$

$$\Delta W_{ji}^{hc} = \eta(T(k) - O_l(k)) f'(O_l(k)) V_{lj} f'(H_j(k)) u_i(k) \quad (8)$$

$$\Delta \beta_c = \eta(T(k) - O_l(k)) f'(O_l(k)) V_{lj} f'(H_j(k)) W_{jc}^{hc} f'(M_c^{co}(k)) M_c^{co}(k-1) \quad (9)$$

$$\Delta \zeta_c = \eta(T(k) - O_l(k)) f'(O_l(k)) V_{lj} f'(H_j(k)) W_{jc}^{hc} f'(M_c^{co}(k)) H_j(k-1) \quad (10)$$

TRAINING ALGORITHM

The training algorithm that will be used for training MRN is very simple and is based on the well-known Backpropagation (BP) training algorithm.

The general steps are:

1. Initialisation step.
   a. Choose error goal (*eg*).
   b. Set the learning rate (η).
   c. Initialise all weights (W_{jc}^{hc}, W_{ji}^{hc} and V_{lj}) to small random values between (0.5, -0.5).
   d. Set the number of the training data samples (K).
   e. Set the sum of the square error to zero (sse=0).
   f. Let $k \longleftarrow 1$
   g. Choose the number of input nodes (I).

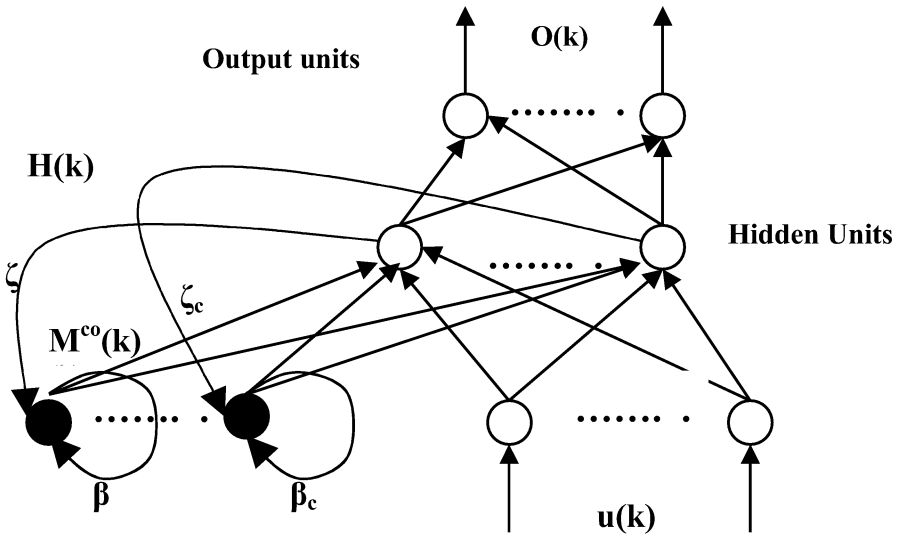


Figure 2. The MRN.

h. Choose the number of hidden nodes (J).
i. Choose the number of context nodes (C).
j. Choose the number of output nodes (L).

2. Choose the type of the activation function $f(.)$.

3. Set $H_j(0) = M_c^{co}(0) = 0.5$ for all (j=1,2…J) and (c=1,2…C).

4. Present the input signal $u_i(k)$ and let

$$M_c^{co} = f\left[\beta_c M_c^{co}(k-1) + \zeta_c H_j(k-1)\right]$$

5. Compute the hidden layer output as:

$$H_j = f\left[\sum_{c=1}^{C} W_{jc}^{hc} M_c^{co}(k) + \sum_{i=1}^{I} W_{ji}^{hu} u_i(k) + W_j^b\right]$$

where W_{jc}^{hc}, W_{ji}^{hu} are weights from context and input nodes to hidden nodes. W_j^b is the bias weight vector of the hidden layer.

6. Compute the output layer output as;

$$O_l(k) = f\left[\sum_{j=1}^{J} V_{lj} H_j(k) + V_l^b\right]$$

where V_{lj} and V_l^b are the weight and the bias of the output layer.

7. Compute:

e=T(k)-O(k)
sse=sse+(e*e)

where e is the error and T(k) is the desired output.

8. Compute the delta signals $\partial_{ol}, \partial_{hj}, \partial_c$ that correspond to the output, hidden and context layers respectively.

$$\partial_{ol} = f'\left(O_l(k)\right)e$$

$$\partial_{hj} = f'\left(H_j(k)\right)\sum_{l=1}^{L} \partial_{ol} V_{lj}$$

$$\partial_c = f'\left(M_j^{co}(k)\right)\sum_{j=1}^{J}\sum_{c=1}^{C} \partial_{hj} W_{jc}^{hc}$$

9. Adjust the weight of the output layer as:

$$V_{lj} \leftarrow V_{lj} + \eta\partial_{ol} H_j$$

$$V_l^b \leftarrow V_l^b + \eta\partial_{ol}$$

10. Adjust the weights of the hidden layer as:

$$W_{ji}^{hu} \leftarrow W_{ji}^{hu} + \eta\partial_{hj} u_i(k)$$

$$W_{jc}^{hc} \leftarrow W_{jc}^{hc} + \eta\partial_{hj} M_c^{co}(k)$$

$$W^b \leftarrow W^b + \eta\partial_{hj}$$

11. Adjust the weights of the context layer as:

$$\beta_c \leftarrow \beta_c + \eta\partial_c(c) M_c^{co}(k-1)$$

$$\zeta_c \leftarrow \zeta_c + \eta\zeta_c(c) H_j(k-1)$$

12. Let

$$M_c^{co}(k-1) = M_c^{co}(k)$$

$$H_j(k-1) = H_j(k)$$

13. If $k \langle K ... then ... k \leftarrow k+1$ and go to step 4
Else go to step 14.

14. The training cycle is completed. Compute the RMS of the error as:

$$RMS = \sqrt{\frac{see}{K}}$$

15. If $RMS \le eg ... then$ terminate the training and save all the computed weights.
Else set $see \leftarrow 0 ... and .... k \leftarrow 1$ and initiate new training cycle by going to step 3.

SYSTEM IDENTIFICATION USING MRN

The parallel identification scheme is the simplest among the system identification methods that are based on NNs. The configuration of this scheme is shown in figure (3). In this identification method, the NN works gradually to learn the model of the plant under test. The training algorithm developed in this paper is used to train the NN so that proper weights of the trainable connections can be produced. The aim of the learning is to minimize the RMS of the error with minimum number of applied epochs and minimum computation time.

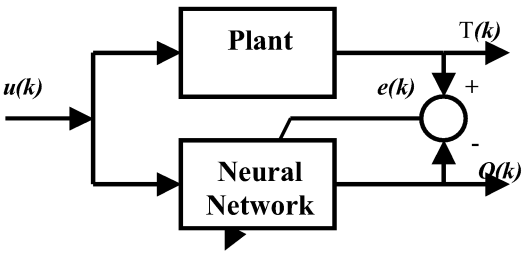


Figure 3. Parallel Identification Model.

SIMULATION RESULTS

In this section, the performance of the MRN in system identification will be examined. One linear and one nonlinear system will be considered. It was assumed that

these systems are bounded input and bounded output stable. The response of the MRN is then compared with the response of the FFNN and MELN. All of the three structures have single input node, six nodes in the hidden/context layers and one output node. Software has been written in MATLAB language to implement the networks explained earlier.

Example 1- Identification of Second-Order linear Plant:

The plant is described by the equation:

$$y(k+1) = 0.4y(k) + 0.2y(k-1) + 0.3u(k) + 0.6u(k-1)$$

The parallel model was used to identify the given plant. During the training phase, random signal was presented to the input. Meanwhile, during the test phase, different signal was used.

Simulation results are shown in figure (4). It is clear, figure (4-a), that the MRN reached the required error goal faster than the other two networks.

The responses of the plant and the three networks to the same input signal are plotted in figures (4-b) to (4-d). As expected, the MRN gives better response than the other two networks. The adaptation processes of the context weights are illustrated in figures (4-e) and (4-f).

Example 2- Identification of a Second-Order Nonlinear Plant:

In the second example a more complex plant was chosen for identification. The plant in this case, which is a second order nonlinear system, is described by the following difference equation:

$$y(k+1) = 0.35\left[\frac{y(k-1)y(k)\big(y(k)+2.5\big)}{1+y(k)^2+y(k-1)^2} + u(k)\right]$$

As in example 1 the parallel model was used to identify the given plant using same training and test signals.

Figure (5) demonstrates the simulation results. Once again, it is clear from figure (5-a) that the MRN reached the required error goal faster than the other two networks.

The responses of the plant and the three network structures to the same input signal are illustrated in figures (5-b) to (5-d). As expected, the MRN gives better response than the other two network structures. Figures (5-e) and (5-f) show the adaptation process of the context unit's weights.

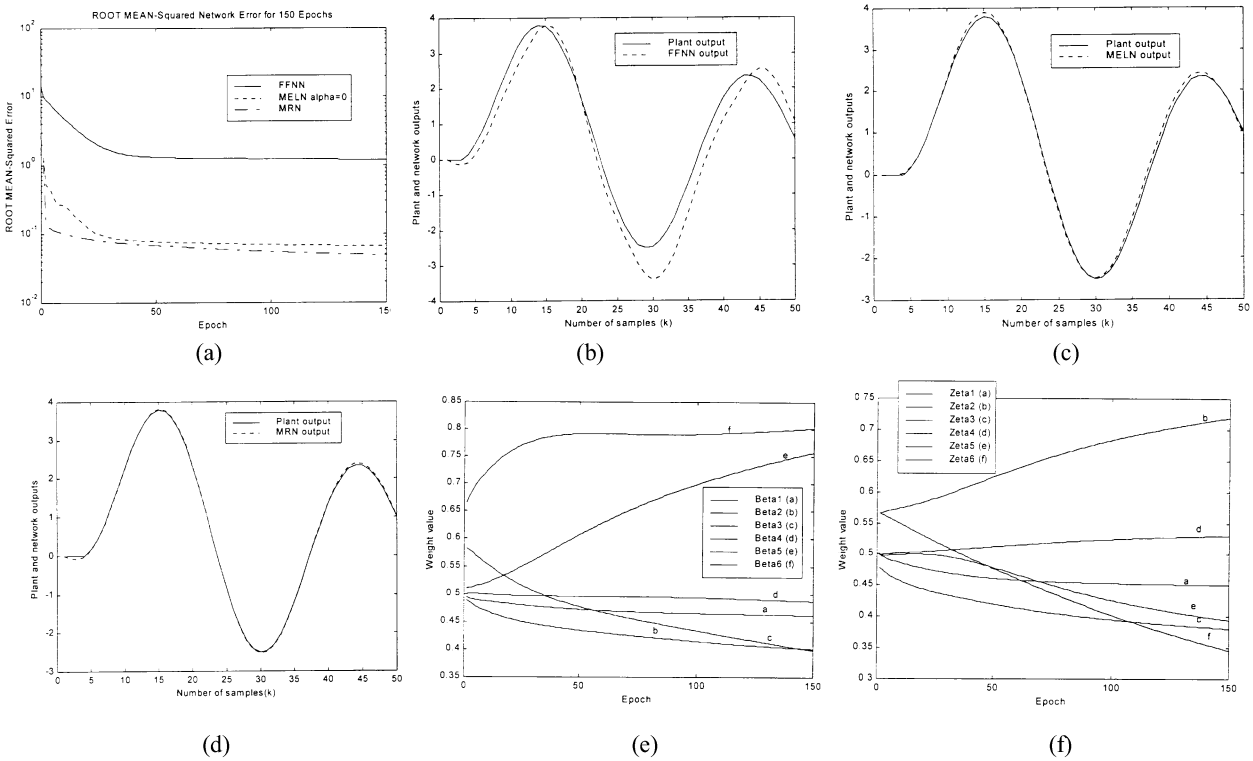


Figure 4. Simulation Results of Example 1.

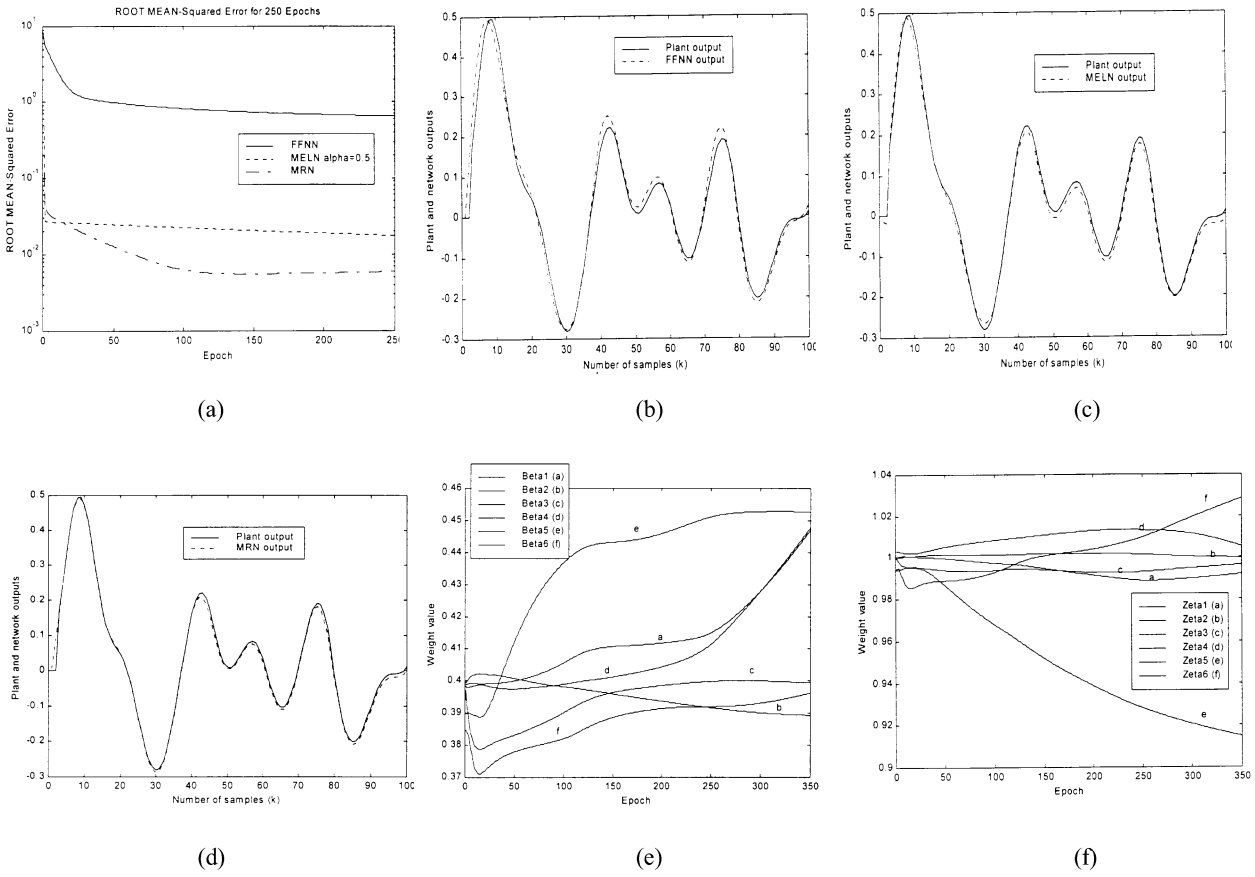


Figure 5. Simulation Results of Example 2.

CONCLUSIONS

A training algorithm for a modified structure of PRNNs is proposed in this paper. The training of the MRN is simple and based on the standard BP training algorithm. The MRN overcome most of the disadvantages of the PRNNs that have fixed self-connection and fixed feedback-connection.

The performance of the MRN was simulated, tested and compared with that of the MLFFNN and MELN. The three neural structures were used in the identification process of linear and nonlinear plants. Simulation results proved that the MRN reach the error goal much faster than the MLFFNN and the MELN.

REFERENCES

Chen, F. and H.K. Khalil. 1992. "Adaptive Control of Non-Linear Systems Using Neural Networks". *Int. Journal of Control*, Vol.55, No.6, 1299-1317.

Elman, J.L. 1990. "Finding Structure in Time." *Cognitive Science*, 14, 179-211.

Elman, J.L. 1991. "Distributed Representations, Simple Recurrent Networks, a Grammatical Structure." *Machine Learning*, 7, 195-225.

Kanjilal, P.P. 1995. *Adaptive Prediction of Predictive Control.* IEEE Control Engineering. Peter Peregrinus Ltd.

Kothari, R. and K. Agyepong. 1997. "Induced Specialization of Context Units for Temporal Pattern Recognition and Reproduction". In *Neural Networks for Signal Processing VII, Proceedings of the 1997 IEEE. Signal Processing Society Workshop*, 131-141.

Kwok, D.P.; P. Wang; and K. Zhou. 1994. "Process Identification Using a Modified Elman Neural Net". In *International Symposium on Speech, Image Processing and Neural Networks* (Hong Kong, April 13-16). IEEE, 499-501.

Pham, D.T. and X. Liu. 1997. *Neural Networks for Identification, Prediction and Control.* Spriner-Verag , New York.

Sastry, B.S.; G. Santharan; and K.P. Unnikrishnan. 1994. "Memory Neural Networks for Identification and Control of Dynamical Systems". *IEEE Transc. On Neural Networks*, Vol.5, No.2, (March), 306-319.

Santos, J. and R.J. Duro. 1998. "Learning to Reason in Time Using Discrete Time Backpropagation". In Proceeding of the *International Conference on Neural Networks and Brain.*

DESIGN OF NEURO-FUZZY CONTROLLER FOR WATER-LEVEL TANK UTILIZING GENNETIC ALGORITHMS

Mohammed Z. AL-FAIZ, Ph.D.,
Head of Computer Eng. Dept.
College Of Eng./Al-Nahrain Univ.
E-mail: mzalfaiz@yahoo.com

Sawsan Kamal, Ph.D.,
Computer Science Dept.
College Of Science/Al-Nahrain Univ.
E-mail: smzozo@hotmail.com

ABSTRACT

Fusion of Artificial Neural networks (ANNs) and Fuzzy Inference Systems (FIS) have attracted the growing interest of researchers in various scientific and engineering areas due to the growing needs of adaptive intelligent systems to solve real world problems. In this paper, Fuzzy Logic (FL), Neural Network (NN), and Genetic Algorithms (GAs) were combined to design and tune a neuro-fuzzy controller (NFC). This design is based on multi-layer neuro-fuzzy network. The adaptation of neuro-fuzzy rules consequents and tuning of NFC weights is accomplished utilizing genetic algorithms. Then ineffective rules are removed from the rule-base of the controller. A real code representation is used to encode the GA chromosome. Two selection methods are used, namely, Roulette wheel and Tournament selection methods. The steps of building, tuning, and the removal of the ineffective rules are accomplished in an off-line phase. In on-line phase, the resulting NFC is operated and it is noticed that the response of the system is not robust enough. So, to avoid any shortage that my happen in the system performance, an adaptive fuzzy controller is added to the NFC. Application of the NFC to a water-level tank is investigated in this paper.

Key Words: Artificial Neural Network, Fuzzy Logic, Genetic Algorithms, Neuro-Fuzzy Controller.

1- INTRODUCTION

Artificial Neural Networks (ANN) have become popular models for a broad variety of modeling and control problems. However, ANN adaptive controllers cannot incorporate linguistic system descriptions, because it is difficult to extract knowledge from them in a comprehensive way. On the other hand, fuzzy inference systems represent knowledge using linguistic labels and may easily be interpreted. Recently, neuro-fuzzy approach has become a popular research area. In contrast to pure neural or fuzzy methods, the neuro-fuzzy method possesses both of their advantages the learning and adaptation capabilities of the neural networks and also providing an inference approach that enables approximate reasoning capabilities (Brown and Harris – 1994).

The integration of artificial neural network (ANN) learning and fuzzy logic (FL) approximate reasoning in one architecture, to overcome individual limitations and achieve synergetic effects through hybridization of these techniques, has in last ten years contributed to a large number of Neuro-Fuzzy (NF) architectures. NF techniques override the classical control methods in many aspects, such as algorithm simplicity, system robustness and the ability to handle imprecision and uncertainty (Abraham, and Nath – 2000)

2- WATER-LEVEL TANK SYSTEM:

As shown in figure 1, a storage tank is used as a buffer water supply and it is replenished from a mains water pipeline. The tank water level is controlled by float-valve mechanism which has frictional effects so that the relationship between the water level and the valve opening is non-linear (Harris - 2000). The relationship is

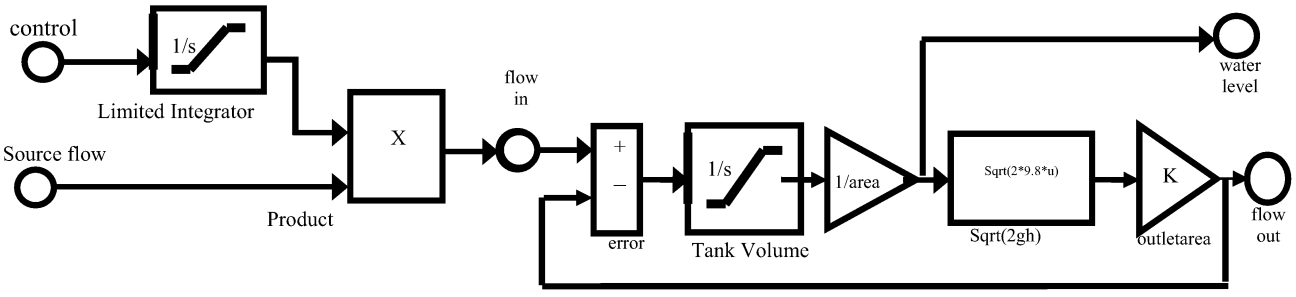


Figure 1: System model (valve+water_tank)

broadly governed by the fact that the valve is fully open when the water level is below 10% of the setpoint, and, is fully closed when it is above the setpoint.

3- NEURO-FUZZY CONTROLLER:

Classical control theory is based on mathematical models that describe the behavior of a plant under consideration. The main idea of fuzzy control on the other hand is to build a model of a human control expert who is capable of controlling the plant without thinking in a mathematical model. The control expert specifies his control actions in the form of linguistic rules. These control rules are translated into the framework of fuzzy set theory providing a calculus which can simulate the behavior of the control expert.

The combination of neural networks and fuzzy controllers to so called ***neuro-fuzzy*** models can help to enhance the performance of the controller by using special learning algorithms. There are several approaches to ***neuro-fuzzy*** systems from which a designer can choose (Nauck and Kruse - 1997).

Neural networks are used to ***tune*** membership functions of fuzzy systems that are employed as decision-making systems for controlling equipments. Although fuzzy logic can encode expert knowledge directly using rules with linguistic labels, it usually takes a lot of time to design and tune the membership functions which quantitatively define these linguistic labels. Neural networks learning techniques can automate this process and substantially reduce development time and cost while improving performance (Fuller - 2000).

4- NEURO-FUZZY CONTROLLER (NFC) STRUCTURE:

The NFC consists of four layers. Figure 2 shows the used neuro-fuzzy controller. Each layer performs a specific function. The first two layers represent the fuzzifier. Third layer represents rule-base (i.e., inference engine); the number of nodes in this layer is equal to the number of rules in the rule-base of the controller. Last layer (fourth layer), represents the defuzzifier, the output from this layer represents the control action.

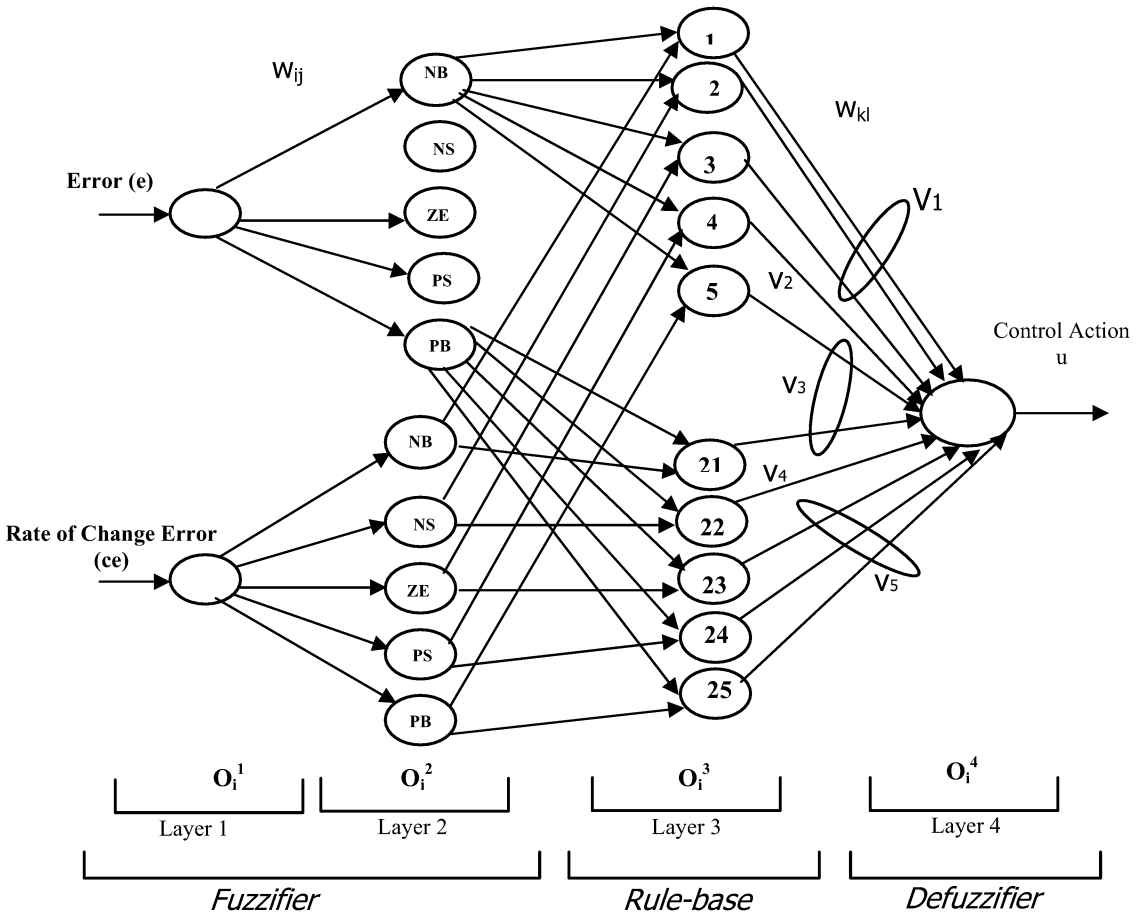


Figure 2: Basic structure of Neuro-Fuzzy Controller.

The controller has two inputs and one output. The input variables are the ***error (e)***, and ***change rate of error (ce).*** The output variable is the ***control action (u)***. Each

variable has five membership functions. They are: Negative Big (NB), Negative Small (NS), Zero (ZE), Positive Small (PS), and Positive Big (PB).

The error is computed as the difference between the reference signal and the output of the system.

$$e(k) = r(k) - y(k) \qquad (1)$$

While the change rate of error is computed as the difference between the current error and the previous one.

$$ce(k) = e(k) - e(k-1) \qquad (2)$$

The error Universe Of Discourse (UOD) for NFC controller is taken as [-1, 1], [-0.04, 0.04] for rate change of error, [-1, 1] for control action.

This structure includes a five by five rule fuzzy controller where the terms shown on the antecedent nodes represent linguistic variables; therefore, the number of rules is five by five which equals to 25-rules.

The term v_j (j=1...5) represents the fuzzy sets describing the consequents or control action.

5- NEURO-FUZZY CONTROLLER BASED ON GENETIC ALGORITHMS (NFG):

The structure of neuro-fuzzy controller, which has been described in figure 2 previously, will be used in the design of NFG controller. GA was used to build this controller, for the properties that GA has in search operations when the data is huge and random.

Neuro-fuzzy controller consists of two main phases:

1- Off-line phase.

This phase includes three steps:

a- Step 1 - Finding the best consequents for connections between layer 3 and layer 4 (see figure 2) of the NFC, i.e., find the suitable consequents to have an acceptable response for the system.

b- Step 2 - Tune the membership function values to get acceptable results (i.e., minimum error).

c- Step 3 - Remove the don't care rules (i.e., the rules that are ineffective during the work of the system all the time).

Points a, and b, are accomplished utilizing GA, while point c is accomplished based on ineffective change on **IAE** (Ogata - 1997) (**Integral Absolute-Error**), which is given as:

$$J = \int_0^\infty |e(t)|dt \qquad \ldots (3)$$

Any GA starts with a population of randomly generated solutions, chromosomes, and advances toward better solutions by applying genetic operators, modeled on the genetic processes occurring in nature. In this work the length of the chromosome is specified according to the problem demands. Then many population sizes were tried to find the suitable one to give a proper solution. The population size was taken as replication of chromosome length which is named ***n***, i.e., 4***n***, 6***n***, 8***n***, and 10***n*** where tried to see their effects on the solution. Two selection methods were used in this work, namely, **Roulette wheel** and **Tournament** selection methods.

2- On-line phase.

An adaptive controller is added, to overcome of non-linearity of the system and to increase system robustness,

A learning control system is designed so that its learning controller has the ability to improve the performance of the closed loop system by generating command inputs to the plant and utilizing feedback information from the plant, the Fuzzy Model Reference Learning Controller (FMRLC) is a direct model reference adaptive controller. The block diagram for the FMRLC is shown in figure 3. It has four main parts; the plant (valve + water tank), the fuzzy controller, the reference model, and the learning mechanism (adaptive fuzzy controller)

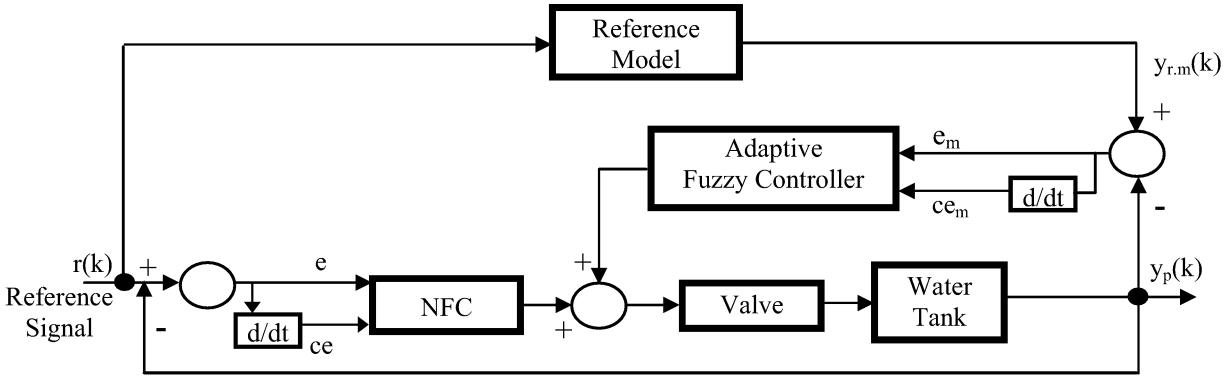


Figure 3: Fuzzy model Reference Learning control.

(Passino and Yurkovich - 1998).

The transfer function of the Reference Model (R.M) is chosen as 2nd order model with ζ =0.9 and ω_n =1.6 rad/sec, and is given as in Equ.4. This model is selected to satisfy the required dynamic performance. The transfer function is:

$$G(s) = \frac{2.56}{s^2 + 2.88s + 2.56} \qquad \ldots (4)$$

The adaptive fuzzy controller has two inputs, error (e_m) and change of error (ce_m), and one output. The error is computed as the difference between the reference model output and controlled plant output. Each variable has five fuzzy sets. So, the rules of adaptive fuzzy ontroller are 25 rules.

$$e_m(k) = y_{r.m}(k) - y_p(k) \quad (5)$$

$$ce_m(k) = e_m(k) - e_m(k-1) \quad (6)$$

The error Universe Of Discourse (UOD) for adaptive controller is taken as [-1, 1], [-0.04, 0.04] for rate change of error, [-1, 1] for control action.

6- SIMULATION RESULTS:

The used training signal is considered as:

$$r(k) = \begin{cases} 1.5 & 0 \le k \le 100 \\ 1 & 100 < k \le 200 \\ 1.7 & 200 < k \le 300 \\ 1.5 & 300 < k \le 400 \end{cases} \quad \ldots . \ (7)$$

where *r(k)* is the training signal and *k* is sample number.

Four cases of initialization were used are:

- Case 1 - Initial rule-base set and condition.
  In this case the first chromosome in the population is equal to the initial rule-base. The rest is generated according to the condition stated above.
- Case 2 - No initial rule-base set and condition.
  In this case, all the chromosomes in the population are generated according to the condition stated above.
- Case 3 - Initial and no condition.
  In this case the first chromosome in the population is equal to the initial rule-base. The rest is generated randomly between 1 & 5.
- Case 4 - No initial and no condition.
  In this case, all the chromosomes in the population are generated randomly between 1 & 5.

Figure 4 represents four cases, which are chosen to make a comparison among them and a rule-base set is chosen to be the result of step 1. This result would be used for step 2 to be tuned.

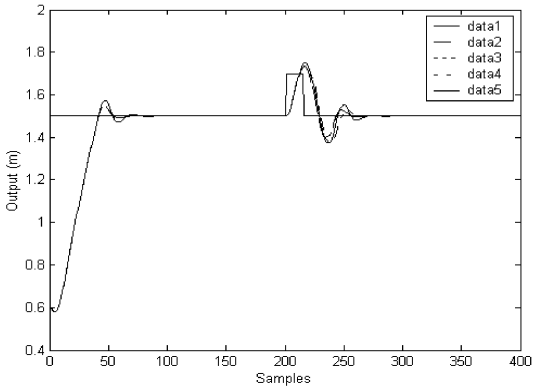


Figure 4: Comparison among four cases results from step 1.

The previous figure represents the selected results from step 1. Data 1 represents the tested signal. Data 2 represents the result of roulette wheel selection method, case 1, training signal 1, and population size 6n. Data 3 represents the result of tournament selection method, case 1, training signal 1, and population size 4n. Data 4 represents the result of roulette wheel selection method, case 1, training signal 2, and population size 4n. Data 5 represents the result of tournament selection method, case 1, training signal 2, and population size 4n.

Table 1 gives the result summary of figure 4. And from which, it can be concluded that data 2, and data 3 are the best. So the rule-base of data 2 is taken as a result from step 1.

Table 1: System response parameters for figure 4.

	t_r (sec)	t_p (sec)	M_P
data 2	4.2	4.7	1.5458
data 3	4.2	4.7	1.5458
data 4	4.2	4.7	1.5458
data 5	4.2	4.7	1.5720

In figure 5, two cases are chosen to make a comparison among them and tuned membership function values are chosen to be the result of step 2. This result would be used for step 3.

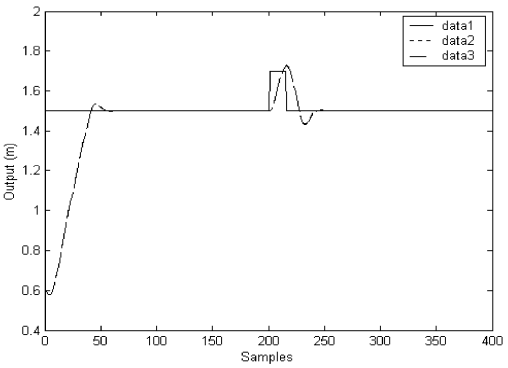


Figure 5: Comparison among tuning results from step 2.

The previous figure represents the selected results from step 2. Data 1 represents the tested signal. Data 2 represents the result of roulette wheel selection method,

and population size 6n. Data 3 represents the result of tournament selection method, and population size 4n.

Table 2: System response parameters for figure 5.

	t_r (sec)	t_p (sec)	M_P
data2	4.2	4.5	1.5341
data 3	4.2	4.5	1.5363

Table 2 gives the result summary of figure 5. And from which, it can be concluded that data 2, and data 3 are the same, but data 3 has greater M_p value. So the tuned membership function values data 2 is taken as a result from step 2.

In step 3, the ineffective rules are removed to minimize the amount of computations for the rule set evaluation. Figure 6 represents the result of step 3, and Table 3 gives the result summary of figure 6.

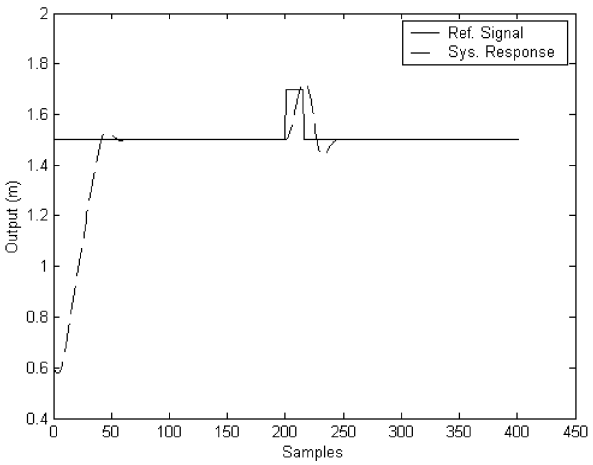


Figure 6: a. System response of step 3.

Table 3: System response parameters for figure 6.

t_r (sec)	t_p (sec)	M_P
4.2	4.5	1.5341

Figure 7 represents the output of the system in on-line phase with and without the operating of the adaptive fuzzy control.

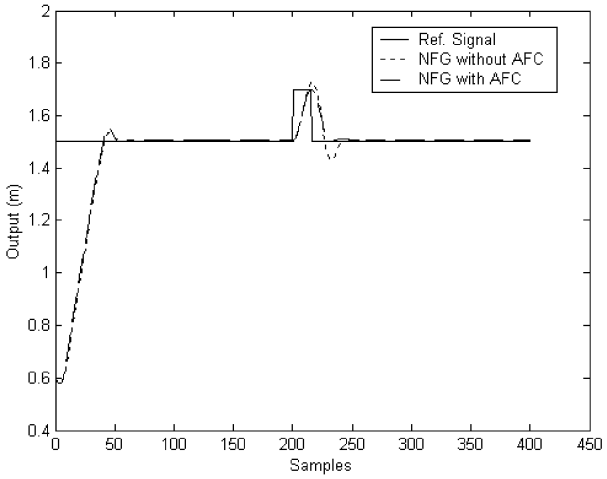


Figure 7: Robustness comparison between response without AFC and with AFC

Table 4: System response parameters for figure 7.

	t_r (sec)	t_p (sec)	M_p
NFG	4.2	4.5	1.5363
NFG with AFC	4.1	4.5	1.5546

From figure 7 and table 4, it can be seen that working with AFC is faster and can respond properly to sudden changes.

7- CONCLUSIONS:

From research results, it can be concluded that:

- The structure of the NFC represents the rules of the fuzzy control exactly. This allows some degree of flexibility in adjusting the fuzzy rules in order to achieve better performance.
- Accurate initial rule-base values are those values that are selected as close as possible to the solution. The choice of these values has a major effect on the response of the system.
- When accurate initial value is used, the increasing of the population size does not affect the results (system response).
- During the attempt to select the best partitioning of the UOD, it was found that tournament selection method is better than roulette wheel selection method when equal partitions of membership functions were applied. On the other hand, if asymmetric partitions were applied, then both of selection methods will give close results, i.e., no remarkable difference is sensed between the results of the two methods.

8- REFERENCES:

– Abraham A., and, Nath B., 2000, " Designing Optimal Neuro-Fuzzy Architectures for intelligent Control", in Proc. of Sixth International Conference on Control, Automation, Robotics and Vision, (ICARCV 2000 - Singapore).

– Brown M., and Harris C. , 1994, " Neurofuzzy Adaptive Modeling and Control ", Printice Hall, New York.

– Fuller R., 2000, " Introduction to Neuro-Fuzzy Systems ", Physica-Verlag.

– Harris j., 2000, " An Introduction to Fuzzy Logic Applications ", Kluwer Academic Publishers, London.

– Nauck D. and Kruse R., 1997," Choosing Neuro-Fuzzy Models ",Proc. Fifth European Congress on Intelligent Techniques and Soft Computing (EUFIT'94), Aachen, pp. 552-557.

– Ogata K., 1997, " Modern Control Engineering ", Prentice-Hall International Editions.

– Passino K. M., and, Yurkovich S., 1998, " Fuzzy Control ", Addison Wesley.

AUTHOR LISTING

AUTHOR LISTING

Adamko N. **30**
Albertos P. **48**
Al-Faiz M.Z. **355**
Al-Faysale M.S.M. **350**
Alonso M. **122**
Al-Qassar A.A. **345**
Amselem G. **53**
André C. **338**
Andreu D. **294**
Anglani A. **35**
Aumeerally M. **306**
Ausborn J. **135/148**
Azil A. **111**

Balsamo S. **206**
Baraza J.C. **239**
Barbu A. **247**
Barhen J. **53**
Barkaoui K. **214**
Bellabdaoui A. **320**
Benítez I.J. **48**
Benyamina D. **43**
Bertelle C. **164/325**
Bertolino S. **153**
Bobeanu C.V. **284**
Bonomo A. **143**
Boukhobza J. **116**
Brito A.E.S.C. **103**
Burmester S. **66**

Castelluzzo L. **35**
Champilomati E. **280**
Coll S. **122**
Collart Dutilleul.S. **265**
Colloc J. **325**
Constantinescu A. **301**
Cotofana S. **5**
Craye E. **265**

Dahan J. **53**
de Andres D. **239**
Delaplace G. **338**
Dewettinck K. **333**
Dhouibi H. **265**
Díez, J.L. **48**
Dohmen L. **188**
Dormido S. **128**
Dumoulin N. **175**
Dupleac D. **301**

Ebecken N.F.F. **143**
Ehling N. **201**
Escheikh M. **214**
Esteban P. **294**

Filipowicz B. **196**
Foubert I. **333**
Fridman A. **25**
Fridman O. **25**

Gambuzza A. **66**
Ghorashi R. **159**
Giese H., ... **66**
Gil P. **239**
Grieco A. **35**
Guérin R. **338**

Hallam N. **43**
Haßlinger G. **201**
Hill D.R.C. **15**
Holobar A. **234**

Iacono M. **220**
Ioualalen M. **272**

Jamali A. **181**

Kamal S. **355**
Karbowski A. **229**
Kavička A. **30**
Keshavarzi-Rooniz N. **159**
Kindler E. **94**
Kirytopoulos K. **280**
Klima V. **30**
Kocik R. **79**
Kohlmeyer J. **140**
Kozakiewicz A. **229**
Křivý I. **94**
Kwiecień J. **196**

Lacomme P. **94**
Lacort J.A. **48**
Lekýr M. **30**
Lemus L. **239**
Lencse G. **89**
Leopoulos V. **280**
Lerebourg S. **164**
Leuhet J.-C. **338**
Levchenkov A. **99**

AUTHOR LISTING

Lian Y.-H. **58**
Licht T. **188**
Luczak H. **188**

Maatoug J. **214**
Mader W. **135**
Malinowski K. **229**
Mamprin R. **206**
Manolescu A. **5**
Marrone S. **220**
Martí A. **122**
Martin C. **128**
Marzbanrad J. **181**
Marzolla M. **206**
Mazzocca N. **220**
Mîndruță C. **74**
Mohammed W.A.F. **345**
Moreau A. **338**
Mor-Yaroslavtsev A. **99**
Moscato F. **220**

Najafi M. **111**
Nariman-zadeh N. **181**
Nassehi V. **159**
Niculiu T. **5**
Nielsen E.H. **313**
Nikoukhah R. **111**
Nketsa A. **294**
Nucci F. **35**

Oberschelp O. **66**
Ojstersek M. **234**
Oleynik A. **25**
Olivier D. **164**

Pascal J.C. **294**
Pasinato S. **153**
Pereira G.C. **143**
Pla M.M. **239**
Prevost G. **164**
Prisecaru I. **301**
Protopopescu V. **53**

Rabouille S. **172**
Randall C. **53**
Ratz D. **252**
Romanin-Jacur G. **153**

Salmi N. **272**
Schmid M. **140**
Schmidt L. **188**
Schmitz P. **188**
Serrano J.J. **122**
Sitte R. **306**
Soetaert K. **172**
Sorel Y. **79**
Sroczan E.M. **257**
Staal M. **172**
Stein W. **135/148**
Stocky W. **252**
Straub O. **135**

Tanguy A. **94**
Teixeira J.M.F. **103**
Thanh C.B. **289**
Timsit C. **116**
Torrez C. **338**
Traczinski D. **201**
Tranouez P. **164**
Tsogkas C. **280**

Urbaniak A. **257**
Urquia A. **128**

Van Landeghem H. **58/284**
Vane Z. **53**
Vanrolleghem P.A. **333**
von Hagen C.-R. **252**

Youssif A.H. **345**

Zacchino S. **35**
Zazula D. **234**